VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

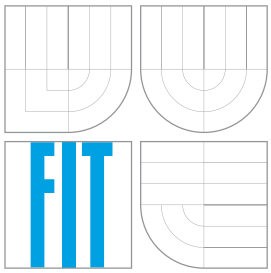
RYCHLÁ DETEKCE DOPRAVNÍCH ZNAČEK V OBRAZE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JAKUB SOCHOR
AUTHOR

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

RYCHLÁ DETEKCE DOPRAVNÍCH ZNAČEK V OBRAZE

FAST DETECTION OF TRAFFIC SIGNS IN IMAGE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAKUB SOCHOR
AUTHOR

VEDOUCÍ PRÁCE — Doc. Ing. ADAM HEROUT, Ph.D.
SUPERVISOR

BRNO 2012

Abstrakt

Tato bakalářská práce se zabývá detekcí dopravních značek v obraze, přičemž cílem bylo zpracování v reálném čase. Nejprve budou popsány algoritmy a způsoby, jakými mohou být dopravní značky detekovány. Následně bude popsána detekce dopravních značek využívající jejich tvarů a úpravy tohoto algoritmu provedené v rámci této práce. V další části bude provedeno vyhodnocení výsledků, kterých se podařilo dosáhnout.

Abstract

This bachelor thesis focuses on detection of traffic signs in real-time. First of all, algorithms used for traffic signs detection will be presented. Description of approach used in this thesis based on shapes of traffic signs and modifications of this algorithm will follow. Evaluation of accomplished results with this algorithm will be also presented.

Klíčová slova

detekce, dopravní značky, Houghova transformace, Cannyho detektor hran, zpracování v reálném čase

Keywords

Detection, Traffic signs, Hough tranform, Canny edge detector, real-time

Citace

Jakub Sochor: Rychlá detekce dopravních značek v obraze, bakalářská práce, Brno, FIT VUT v Brně, 2012

Rychlá detekce dopravních značek v obraze

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Ing. Adama Herouta, Ph.D.

.....
Jakub Sochor
9. května 2012

Poděkování

Rád bych poděkoval vedoucímu práce, doc. Ing. Adamovi Heroutovi, Ph.D, za jeho odbornou pomoc a také celkové vedení k co nejlepším výsledkům. Dále bych chtěl poděkovat Tomášovi Svobodovi za jeho trpělivost při komunikaci ohledně jeho diplomové práce.

Obsah

Seznam obrázků

Kapitola 1

Úvod

V dnešní době se stále zvyšuje spoluúčast počítačových a vestavěných systémů na řízení automobilu. Vozy již umí například detekovat neúmyslné opuštění jízdního pruhu, případně existují další systémy, které usnadňují uživateli vozidla řízení. Tato bakalářská práce se zaměřuje na detekci dopravních značek, přičemž cílem bylo zpracování obrazu v reálném čase.

Systém umožňující detekci a následnou klasifikaci dopravních značek v reálném čase je možné využít ve velkém množství situací. Lze například detekovat maximální povolenou rychlost a varovat řidiče při jejím překročení. Případně je možné zobrazovat další upozornění, která se vyskytla na dopravních značkách.

Detekce dopravních značek implementována a popsána v této bakalářské práci je založena na tvarech těchto dopravních značek. Je tedy využíváno faktu, že dopravní značky mají tvar kruhu, trojúhelníku nebo čtverce.

V kapitole 2 této práce budou nejprve popsány tři odlišné algoritmy, kterých je využíváno pro detekci dopravních značek. Dále budou do detailu představeny v kapitole 3 algoritmy zpracování obrazu a další nutné znalosti, jichž je využíváno v mé práci. V kapitole 4 bude následovat návrh systému detekce dopravních značek, jenž je založen na tvaru těchto značek, a budou prezentovány další vylepšení tohoto způsobu za účelem vyšší rychlosti a větší úspěšnosti detekce. V kapitole 5 bude popsán praktický způsob implementace tohoto systému. Vyhodnocení úspěšnosti a rychlosti implementovaného systému bude popsáno v kapitole 6.

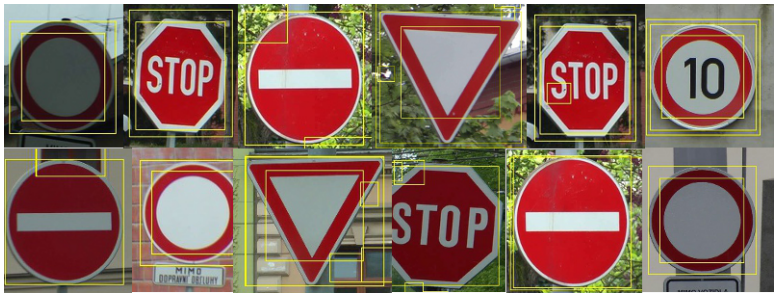
Kapitola 2

Detekce dopravních značek

V této kapitole budou představeny tři cesty a způsoby, jakými je možné provádět detekci dopravních značek. Budou také popsány výsledky, jakých jednotlivé skupiny v rámci své práce dosáhly.

Pro detekci dopravních značek lze využít obecné způsoby pro detekci rigidních objektů jako je například posuvné okno a kaskáda klasifikátoru určující, zdali daný objekt je dopravní značkou či nikoli. Do této kategorie spadá také použití SURF příznaků pro popis objektu. Další možností je využití speciálních vlastností dopravních značek, tedy například toho, že mají předem jasně daný a specifikovaný tvar, jenž je určen zákonem.

Nejprve bude ovšem zařazena detekce dopravních značek do kontextu počítačového vidění a definovány některé pojmy.



Obrázek 2.1: Obrázky dopravních značek

2.1 Detekční úloha, lokalizační úloha, počítačové vidění obecně

V následující části bude nejprve stručně rozebráno počítačové vidění a problémy, které řeší. Dále budou zavedeny některé pojmy, jichž bude v mé práci využíváno. Při psaní této části mé práce jsem vycházel především z knih zabývajících se počítačovým viděním [13], [17] a také z knihy [6].

2.1.1 Počítačové vidění

Linda Shapiro definuje cíl počítačového vidění v knize Computer Vision [13] následujícím způsobem.

Definice 1 *Cílem počítačového vidění je dělat užitečná rozhodnutí o reálných fyzických objektech a scénách na základě snímaných obrázků.*

Počítačové vidění se v současné době zaobírá například rozpoznáváním ručně psaných textů, detekcí neočekávaných překážek na vozovce, analýzou provozu na dálnici, monitorováním bazénu, zdali se aktuálně někdo netopí, nebo například vizuální autentizací.

Takováto použití počítačového vidění se nacházejí ovšem pouze na té nejvyšší úrovni. Pro všechny tyto aplikace je nutné provádět různá zpracování obrazu na nižší úrovni. Mezi operace na nižší úrovni mohou patřit například algoritmy pro následující úkoly.

- Detekce objektů. Detekovat lze například rigidní nebo fluidní objekty, přičemž pro dané typy se velmi často využívá velmi odlišných algoritmů.
- Klasifikace objektů. Klasifikace objektů do předem daných tříd, přičemž detekci objektů lze také zařadit do klasifikace objektů.
- Segmentace. Skupina algoritmů učení bez učitele, jichž je využíváno například pro rozdělení barev v obraze do tříd.
- Rekontstrukce objemových dat. Tyto algoritmy najdou uplatnění například v medicíně, kdy je nutné rekonstruovat 3D model mozku z obrázků, které byly získány pomocí magnetické rezonance.

Detekce dopravních značek v kontextu výše uvedeného rozdělení spadá do detekce rigidních objektů v obraze.

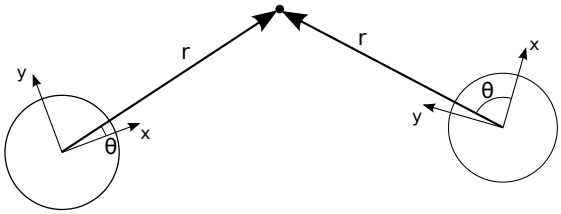
2.1.2 Detekční úloha, lokalizační úloha

Detekční úlohou se rozumí úkol v obraze nalézt, pokud je přítomen, daný objekt a to s co nejmenším počtem nenalezených výskytů a také nejmenším možným počtem falešných detekcí.

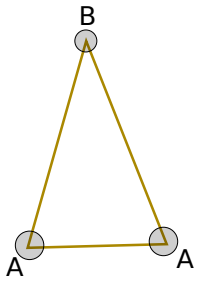
Lokalizační úlohou je chápána snaha detekovaný objekt co nejpřesněji nalézt v obraze, takovým způsobem, aby vzdálenost mezi reálným objektem a jeho označením byla co nejmenší a v ideálním případě se úplně překrývaly. Pro určení přesnosti lokalizace lze využít například metriky F-Measure, která bude popsána dále.

Klasifikační úlohou je myšleno přiřazení daného objektu do jedné z předem daných tříd. Lze uvažovat například klasifikaci obrázku nábytku do podrobnějších tříd, tedy do třídy stůl, židle, skříň, postel atd. Detekční úlohu je možno chápat jako speciální případ klasifikační úlohy, jelikož klasifikujeme do dvou tříd a to „hledaný objekt“ a „něco jiného“.

Dané třídy jsou popisovány pomocí jejich vlastností, kterým lze říkat příznaky. Příznaky, podle kterých je prováděna klasifikace, by měly dané třídy objektů co nejvíce odlišovat.



Obrázek 2.2: Obrázek rozmístění bodů a určení vzdálenosti a úhlu vzhledem k získané orientaci



Obrázek 2.3: Obrázek modelu a jeho bodů (viz tabulka 2.1)

2.2 Detekce dopravních značek využívající vizuálních slov

Během zkoumaní současných algoritmů a způsobů, jak provádět detekci dopravních značek, jsem nalezl mimo jiné také článek [9], ve kterém je popisován algoritmus detekce dopravních značek založený na SURF příznacích [2]. Tento algoritmus se snaží mířit na implementaci ve vestavěných zařízeních, tedy by měl být dostatečně rychlý. Bude následovat popis algoritmu využívaný v tomto článku a výsledky, kterých Toon Goedemé v rámci článku [9] dosáhl.

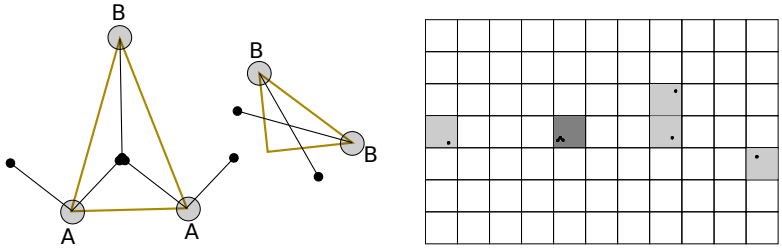
Algoritmus

Nejprve bude popsán algoritmus vytváření modelu a následně bude ukázáno, jak je možné pomocí takovéhoto modelu detekovat dopravní značky. Algoritmus není určen výhradně pro detekci dopravních značek, ale lze s jeho pomocí detekovat také jiné rigidní objekty v obraze.

Vytváření modelu probíhá na obraze obsahující pouze objekt, pro jehož detekci chceme vytvořit model. V tomto obraze jsou určeny SURF příznaky a následně je spočítána jejich orientace pomocí Haarových vlnek, přičemž orientace příznaku je vybrána taková, která maximalizuje odezvu zmíněných Haarových vlnek, jenž jsou postupně natáčeny o úhel $\frac{\pi}{3}$. Poté je nutné pro velké množství těchto SURF příznaku provést jejich segmentaci do skupin a přiřadit jim unikátní identifikátory, přičemž tato segmentace je v rámci článku prováděna pomocí algoritmu k-means. V dalším kroku jsou pro všechny příznaky v daném obraze určeny jejich relativní polohy k pevnému bodu, přičemž tento pevný bod může být například středem objektu, jehož model je vytvářen. Uložen je následně unikátní identifikátor, který byl získán pomocí algoritmu k-means a relativní polohy ke středu objektu, přičemž způsob určování relativní polohy lze vidět na obrázku 2.2. Příklad takovéhoto modelu lze vidět v tabulce 2.1.

Příznak	θ	r
A	1.24	20.25
A	-1.35	22.24
B	0.13	17.23

Tabulka 2.1: Model pomocí SURF příznaků



(a) Obrázek ve kterém je prováděna detekce

(b) Příslušný Houghův prostor

Obrázek 2.4: Obrázek detekce objektu a příslušeného Houghova prostoru

Při následné detekci objektů v obraze je nejprve provedena extrakce SURF příznaků a jejich orientace stejným způsobem jako při vytváření modelu. Dále těmto příznakům jsou přiřazeny unikátní identifikátory, které byly získány při vytváření modelu. Následně jsou určovány body, kde by se mohl nacházet střed detekovaného objektu, k čemuž je využíváno relativní polohy tohoto pevného bodu k SURF příznakům. Pokud je jeden příznak v modelu, vytvořeném při trénování, přítomen vícekrát, jsou uvažovány všechny možné polohy objektu vzhledem k příznaku, ovšem pouze jedna takováto poloha může být správná. Následně je vytvořen Houghův prostor, který je inicializován s nulovými hodnotami, jež jsou zvětšovány, pokud nějaký příznak indikuje, že by v daném bodě měl ležet pevný bod objektu, ke kterému je vztahována poloha příznaků. Příklad lze vidět na obrázku 2.4 s využitím modelu, který byl určen pro objekt na obrázku 2.3.

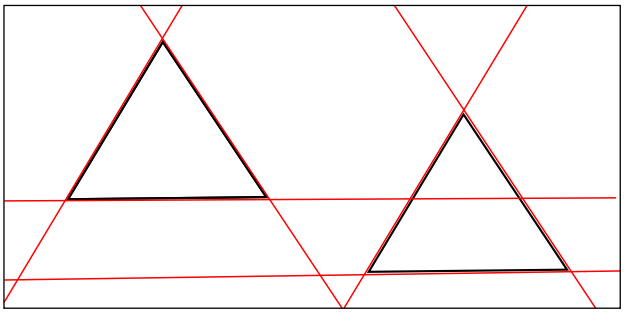
Dosažené výsledky

V rámci článku [9] bylo provedeno testování na 35 obrázcích a získána 82 % úspěšnost detekce, přičemž případy, kdy značky nebyly detekovány, byly způsobeny zejména příliš velkou vzdáleností dopravní značky a v několika případech překrytím jiným objektem.

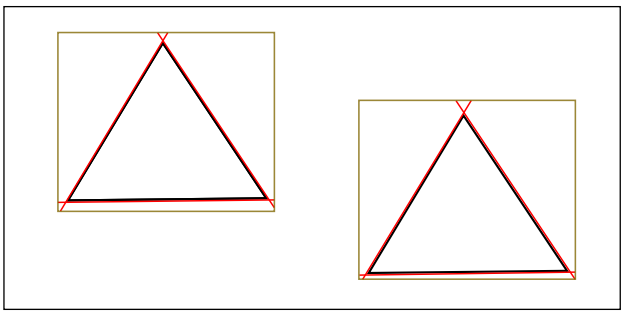
V uvedeném článku nebyla prezentována dosažená rychlost, protože testování bylo prováděno pouze s prototypovou implementací pro software Octave.

2.3 Využití tvaru dopravní značky pro její detekci

V článku [8] je naopak demonstrován diametrálně odlišný přístup k detekci dopravních značek. Algoritmus detekce, dopravních značek popsaný ve výše uvedeném článku, je založen na geometrických vlastnostech a tvaru dopravních značek, jelikož tyto vlastnosti jsou relativně konzistentní po celém světě.



(a) Detekce přímek v celém obraze



(b) Detekce přímek pouze v části obrazu

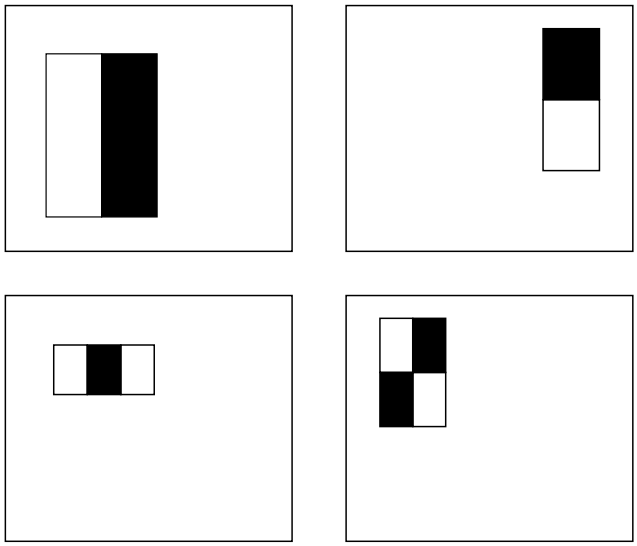
Obrázek 2.5: Rozdíl mezi počtem přímek a průsečíků a především trojúhelníků, pokud je prováděna detekce geometrických primitiv v celém obraze nebo pouze v jeho části.

Algoritmus

Detekce značek tímto algoritmem se tedy snaží v obraze detekovat kružnice, čtverce a trojúhelníky, které patří dopravním značkám. Průběh detekce pomocí tohoto algoritmu je následující.

Nejprve je provedena detekce hran v obraze, přičemž v článku je používán mírně upravený Cannyho detektor hran. Zmíněná úprava spočívá v tom, že jednotlivé prahy pro prahování s hysterzí, které je prováděno v rámci Cannyho detektoru hran, jenž je podrobně popsán v kapitole 3.2.1, je určováno dynamicky v závislosti na rozložení histogramu. Následně jsou detekovány obrysy v obraze, a pokud obrys splňuje kritéria pro to být obrysem dopravní značky, je předán dále. Obrys dopravní značky musí například být uzavřený nebo téměř uzavřený, případně ohraničující obdélník nesmí mít příliš odlišnou výšku a šířku.

V další fázi detekce je nutné nejprve určit pro každý obrys ohraničující obdélník, protože se následně v této oblasti budou detekovat kružnice a přímky. Detekce kružnic a přímek je v článku prováděna pomocí Houghovy transformace. Pokud jsou detekovány přímky v daném výřezu obrazu, je otestováno, jestli lze z nich sestrojit trojúhelník nebo čtverec. Je-li nalezen některý ze zmiňovaných geometrických útvarů, ať už čtverec, trojúhelník nebo kružnice, je výřez považován za potenciální dopravní značku. Provádění Houghovy transformace pouze ve výřezu z obrazu má zejména dva přínosy. Jeden z nich je, že není nutné provádět tuto, časově relativně náročnou operaci, na velkém prostoru a je detekce rychlejší. A také při vytváření trojúhelníků a čtverců z přímek není nutné uvažovat tolik kombinací pro sestrojení daného objektu, které by velmi často byly nesmyslné. Tento problém demonstruje obrázek 2.5.



Obrázek 2.6: Příklad typů příznaků, jež jsou využívány pro posuvné okno

Dosažené výsledky

Pomocí toho algoritmu byli autoři článku [8] schopní dosáhnout detekce dopravních značek v reálném čase. Úspěšnost detekce byla testována na dvou typech dopravních značek, přičemž jedna byla trojúhelníková a druhá kruhová a testovací sada obsahovala téměř 750 dopravních značek. Úspěšnost detekce na této testovací sadě byla v průměru přibližně 95 %.

2.4 Detekce dopravních značek pomocí posuvného okna

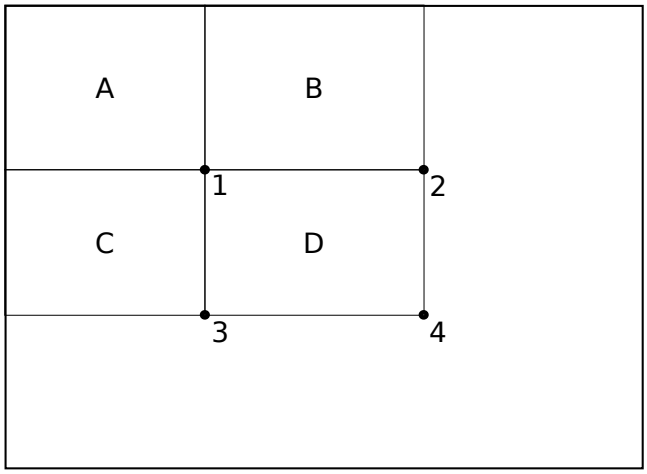
Další možností, jak provádět detekci objektu, jenž nemusí býti pouze dopravní značkou, ale obecným rigidním objektem, je nastíněn v článku [18]. Jedná se o využití slabých klasifikátorů, které jsou přítomny v kaskádě a dohromady tvoří silný klasifikátor. Nejprve bude popsán algoritmus, jak jej autoři článku [18] publikovali, a následně budou představeny úpravy, které udělali autoři článku [1], a prezentovány výsledky, jichž se jim podařilo dosáhnout.

Příznaky

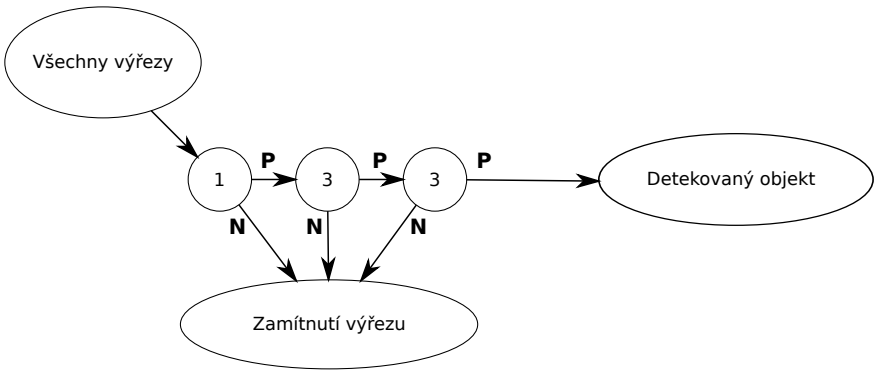
Pro detekci objektu jsou využívané příznaky, které se získávají přímo z obrazu pomocí sčítání a odčítání hodnot bodů. Na obrázku 2.6 lze vidět různé typy takovýchto příznaků, přičemž platí, že pro daný příznak je určen součet bodů nad bílým a černým obdélníkem, a tím získána hodnota jednotlivých obdélníků. Následně je hodnota bílého obdélníku odečtena od hodnoty černého obdélníku a tímto způsobem získán příznak. Takových to příznaků je na výřezu, jenž má 24×24 bodů, 180 000.

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{2.1}$$

Jelikož by výpočet všech součtů pod jedním obdélníkem byl velmi časově náročný a navíc by se tyto výpočty velmi často opakovaly pro stejné body, je využíváno integrálního



Obrázek 2.7: Pomocí integrálního obrazu může být jednoduše spočítán součet bodů ve obdélníku D. Hodnota integrálního obrazu v bodě 1 je rovna součtu bodů v A, pro bod 2 je to součet $A + B$, 3 je naopak určeno jako $A + C$ a 4 obsahuje součet všech obdélníků, tedy $A + B + C + D$. Součet bodů v D je tedy určen výrazem $4 + 1 - 2 - 3$.



Obrázek 2.8: Kaskáda klasifikátorů. Tento příklad obsahuje pouze tři stupně kaskády, ovšem většinou těchto stupňů je řádově více.

obrazu, s jehož pomocí je možné součet bodů pod každým obdélníkem v obraze spočítat v konstantním čase. Integrální obraz má stejné rozměry jako původní a pro každý bod integrálního obrazu o souřadních (x, y) platí rovnice 2.1. Jakým způsobem lze následně získat součet bodů pod obdélníkem obrazu lze vidět na obrázku 2.7.

Algoritmus

Při následné detekci je každý možný výřez obrazu, ve kterém chceme detekovat daný objekt, předán kaskádě klasifikátorů, přičemž každý stupeň této kaskády vyhodnotí dané příznaky, které byly získány při trénování, a pokud výřez vyhoví pravidlům, definovaným v této kaskádě, je předán ke zpracování dalšímu stupni. Pokud výřez kterémukoli stupni nevyhoví, je zahozen, a objekt v tomto výřezu nebyl detekován. Výřezy, které se dostanou až na konec kaskády těchto klasifikátorů, jsou prohlášeny za detekovaný objekt.

Dosažené výsledky

V rámci článku [1] bylo využíváno mírně pozměněného výše popsaného algoritmu pro detekci dopravních značek. Provedené změny spočívají zejména ve využití obecnějších příznaků než na obrázku 2.6, přičemž způsob zpracování je obdobný. Je využíváno větší variace rozmístění černých a bílých obdélníků. Jelikož je takovýchto příznaku ve výřezu 2^{30}, tak pro trénování kaskády klasifikátorů je v článku využíváno evolučních algoritmů.

Ve zmíněném článku byla dosažena a prezentována přibližně 90 % úspěšnost detekce, přičemž bylo možné detekovat i dopravní značky v extrémních světelných podmínkách. V článku nebyla popsána rychlost, jíž se podařilo dosáhnout.

Kapitola 3

Použité algoritmy

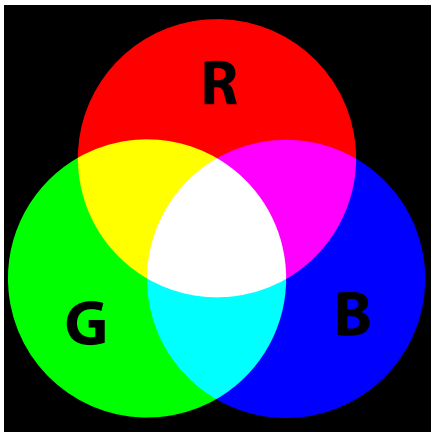
V předchozí kapitole byly představeny způsoby, jak lze provádět detekci dopravních značek. V této kapitole budou popsány reprezentace obrazu a algoritmy, jež jsou důležité a využívané v mé práci.

3.1 Barevné modely

V této části budou popsány tři různé modely pro reprezentování barev v obraze a následně diskutovány jejich výhody a nevýhody. Jednotlivé barevné modely jsou důležité v různých aspektech mé práce.

Při psaní této kapitoly jsem vycházel především z knihy zabývající se počítačovou grafikou [20].

3.1.1 RGB



Obrázek 3.1: RGB aditivní skládání[1]

Barevný model RGB je jeden z nejpoužívanějších barevných modelů, jenž je využíván například v monitorech a obdobných zobrazovacích zařízeních. Barva jednoho bodu v obraze je určena intenzitou základních barevných složek, mezi které patří červená, zelená a modrá

[1] Obrázek byl převzat ze stránek `http://en.wikipedia.org/`

(a) Barevný model RGB (b) Šedotónový obraz

Obrázek 3.2: Porovnání barevného a šedotónového obrazu

barva. Další možné barevné kombinace se získávají aditivním mícháním těchto barev, jak lze vidět na obrázku 3.1. Tedy například pro bílou barvu musí být přítomny všechny základní barvy v plné intenzitě a naopak pro barvu černou nesmí být přítomna žádná z těchto tří barev.

Tento barevný model je velmi často využíván, poněvadž velká množina formátů, se kterými pracuje obraz nebo video, jej využívá pro uložení informace o barvě. Barevný model RGB má ovšem jednu velkou nevýhodu při zpracování obrazu a určování barvy bodu. Zmíněný problém je způsoben zejména tím, že jas barvy není explicitně oddělen od jeho odstínu. Tento model se ukázal tedy nevhodný pro použití v mé práci pro určování barvy bodů a testování, jestli daný bod je červený nebo modrý.

3.1.2 Šedotónový obraz

Obraz ve stupních šedi je velmi často využíván pro jeho jednoduchost, poněvadž obsahuje pro každý bod pouze jednu hodnotu a nikoli tři jak je tomu u RGB modelu. Lze tedy obraz chápat čistě jako dvojrozměrnou matici čísel. Své uplatnění nalezne například v použití spolu s některými detektory hran. Srovnání baravného obrazu s obrazem ve stupních šedi lze vidět na obrázku 3.2.

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{3.1}$$

Pro konverzi z obrazu reprezentovaného pomocí RGB barevného modelu do obrazu v odstínech šedi je využíváno rovnice 3.1, přičemž Y je výsledná hodnota šedotónového pixelu a R, G, B jsou hodnoty RGB barevného modelu. Jednotlivé složky mají různou váhu, protože lidské oko je různě citlivé na základní barevné složky RGB modelu.

3.1.3 HSV

Barva jednoho bodu je pomocí modelu HSV, obdobně jako u modelu RGB, určena třemi složkami. V případě tohoto modelu mají uvedené složky naprosto odlišný význam. Barva bodu je určena pomocí barevného tónu H, sytosti barvy S a jasu V. Barevný tón udává, která barva převládá, sytost barvy určuje příměs šedé barvy, přičemž pokud je sytost barvy minimální, tak se jedná o šedou barvu, a konečně jas definuje, jak moc je daná barva světlá nebo tmavá.

$$V = \max(R, G, B) \tag{3.2}$$

$$S = \begin{cases} \frac{V - \min(R,G,B)}{V} & V \neq 0 \\ 0 & V = 0 \end{cases} \tag{3.3}$$

$$H = \begin{cases} \frac{60(G-B)}{V - \min(R,G,B)} & V = R \\ \frac{120+60(B-R)}{V - \min(R,G,B)} & V = G \\ \frac{240+60(R-G)}{V - \min(R,G,B)} & V = B \end{cases} \tag{3.4}$$

Jednotlivé rovnice pro převod z RGB barevného modelu do modelu HSV jsou označeny 3.2, 3.3 a 3.4, přičemž hodnoty R, G a B jsou jednotlivé složky RGB barevného modelu. Jak si lze všimnout není transformace všech bodů obrazu do tohoto barevného modelu jednoduchá a vyžaduje podstatně více operací, než například převod do obrazu ve stupních šedi.

Tento model se velmi často využívá pro oddělení informace o barvě od informace o jasové složce daného bodu. Toho lze výhodně využít při zkoumání barvy jednotlivého bodu, jelikož lze při zkoumání vynechat jasovou složku, která pak nebude ovlivňovat danou barvu.

3.2 Detekce hran v obraze

V rámci mé práce je velmi důležité provádět kvalitní detekci hran, jak bude ukázano později. Z tohoto důvodu budou představeny dva algoritmy pro detekci hran, kterých je využíváno v mé práci.

3.2.1 Cannyho detektor hran

Cannyho detektor hran, jehož autorem je John Canny, patří k jednomu z nejpoužívanějších detektorů hran, jelikož se snaží splňovat podmínky pro dobrou detekci, které John Canny definoval a budou popsány dále. Informace o Cannyho detektoru hran jsem čerpal zejména z článku Johna Cannyho [3] a také z knihy Milana Sonky o zpracování obrazu a počítačovém vidění [14].

John Canny v rámci svého článku, kde představil popisovaný detektor hran, stanovil tři podmínky, které by měly platit pro detektory hran a jichž se snaží držet. Pokud detektor hran splňuje tyto podmínky, měl by být schopen produkovat relativně dobré výsledky. Podmínky jsou popsány v následujících bodech.

1. Kvalitní detekce. Měla by existovat malá pravděpodobnost selhání v detekci hrany a také malá pravděpodobnost detekce falešné hrany.

2. Kvalitní lokalizace. Body označené jako hrana by měly být co nejblíže ke středu skutečné hrany.

3. Pouze jedna odezva na jednu hranu. Jedna skutečná hrana v obraze by neměla vygenerovat více než jednu hranu.

Jelikož Cannyho detektor hran je koncipován spíše pro obecný signál, který může být jednorozměrný nebo dvourozměrný, tak je pomocí tohoto detektoru hran možné hledat

hrany pouze v obraze s jedním kanálem. Z tohoto důvodu se pro tento úkol většinou využívá obrazu v odstínech šedi, poněvadž je to jeden z nejpřímějších způsobů, jak získat z barevného obrazu se třemi kanály obraz s jedním kanalám. Algoritmus může být popsán v následujících bodech.

1. Provést konvoluci obrazu f s gaussovým filtrem G, dále označováno jako $G * f$.
2. Určit směry normálových vektorů ve všech bodech obrazu pomocí rovnice 3.6.
3. Nálezt správnou polohu hran pomocí operace „potlačení nemaximálních hodnot“ (non-maximal suppression), jenž je vyjádřena pomocí rovnice 3.5. Jelikož konvoluce a derivace jsou asociativní, tak lze nejdřív provést konvoluci $G * f$ a až následně provést derivaci podle směrů $\vec{n}$ určených v bodě 2.
4. Spočítat sílu hrany, tedy určit velikost gradientu s využitím rovnice 3.8, přičemž hodnota G_n je popsána rovnicí 3.7.
5. V obraze provést prahování s hysterzí. Pro toto prahování je nutné určit dva prahy, jeden vyšší T_1 a druhý nižší T_2. Pokud je velikost odezvy hrany v nějakém bodě větší než práh T_1, je tento bod definitivně označen jako hrana. Pokud je velikost odezvy menší než práh T_1, ale větší než práh T_2 a zároveň je tento bod spojen s bodem, který měl tuto odezvu větší než práh T_1, tak je daný bod také označen jako hrana. Všechny ostatní body nejsou označeny jako hrana.

$$0 = \frac{\partial^2}{\partial \vec{n}^2} G * f \tag{3.5}$$

$$\vec{n} = \frac{\nabla (G * f)}{|\nabla (G * f)|} \tag{3.6}$$

$$G_n = \frac{\partial G}{\partial \vec{n}} \tag{3.7}$$

$$|G_n * f| = |\nabla (G * f)| \tag{3.8}$$

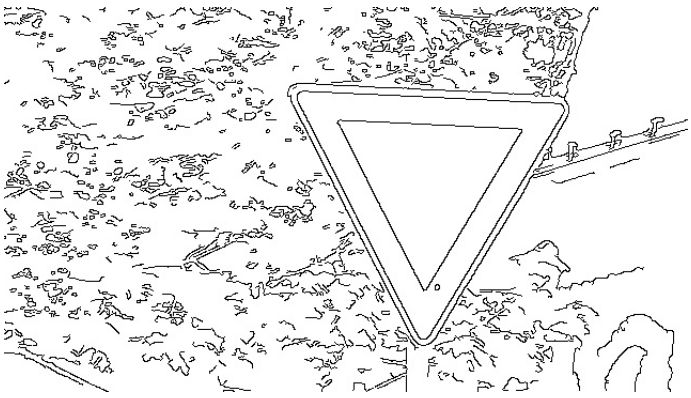
Pro výše popsaný algoritmus existují také jisté jeho variace. Například je možné provádět popsané body vícekrát po sobě se zvyšující se škálou σ symetrického gaussovského filtru G a přidávat do výsledného obrazu hrany, jež nebyly detekovány s menším σ. Obrázek před a po aplikování Cannyho detektoru hran lze vidět na obrázku 3.3 a jak již bylo řečeno, je nutné provádět detekci hran tímto detektorem na obraze, který má pouze jeden kanál, v tomto případě je tedy použit obrázek v odstínech šedi.

3.2.2 Detekce hran v RGB barevném prostoru

Algoritmus pro detekci hran popsaný v článku [19] využívá celého barevného prostoru RGB pro detekci hran. Ovšem na rozdíl od detektorů hran využívající pouze Euklidovskou vzdálenost dvou vektorů, využívá také úhel mezi vektory pro určení rozdílu mezi body a tedy detekci hran. Ve zmíněném vědeckém článku jsou také popisovány problémy těchto jednotlivých metrik a jejich nevýhody, pokud jsou využity samostatně.

Pokud využijeme pouze úhel mezi vektory, tak dokážeme dosáhnout velmi dobrých výsledků u bodů, které mají odlišný barevný tón, avšak pokud využíváme barevný model RGB, tak narazíme na problém u bodů s nízkou intenzitou složek tohoto barevného prostoru, jelikož i malé změny v těchto hodnotách mohou znamenat velkou změnu barevného

(a) Orignální obrázek



(b) Detekované hrany

Obrázek 3.3: Výsledek detekce hran pomocí Cannyho hranového detektoru

tónu. Tyto problémy se nevyskytují při využití Euklidovské vzdálenosti jako metriky pro určení rozdílu bodů, ovšem tato metrika není na druhou stranu schopná správně postihnout body, které mají téměř totožný barevný odstín.

$$D(\vec{v}_1, \vec{v}_2) = ||\vec{v}_1 - \vec{v}_2|| \tag{3.9}$$

$$D(\vec{v}_1, \vec{v}_2) = \sqrt{(v_{1,1} - v_{2,1})^2 + (v_{1,2} - v_{2,2})^2 + (v_{1,3} - v_{2,3})^2} \tag{3.10}$$

Euklidovská vzdálenost mezi dvěma vektory $\vec{v}_1$ a $\vec{v}_2$ je definována rovnicí 3.9, což pro trojrozměrný vektorový prostor znamená přepsání na vztah 3.10. Úhel θ, případně hodnota funkce cosinus tohoto úhlu, mezi vektory je určen pomocí rovnice 3.11. Ve zmíněném článku je ovšem hodnota cosinu úhlu prezentována jako nepříliš vhodná, protože pokud chceme detekovat i nepatrné změny barevného odstínu, které vedou k nepatrné změně úhlu θ, tak funkce cosinus se příliš nemění pro malé hodnoty. Funkce sinus na druhou stranu pro malé hodnoty úhlu roste relativně rychle, tak lze využít známého vzorce pro goniometrické funkce 3.12 a místo funkce cosinus využít funkci sinus, která má již lepší vlastnosti, což vede na rovnici 3.13.

$$\cos\theta = \frac{\vec{v}_1^T \vec{v}_2}{||\vec{v}_1|| \cdot ||\vec{v}_2||} \tag{3.11}$$

$$1 = \sin^2\theta + \cos^2\theta \tag{3.12}$$

$$\sin\theta = \sqrt{1 - \left(\frac{\vec{v}_1^T \vec{v}_2}{||\vec{v}_1|| \cdot ||\vec{v}_2||}\right)^2} \tag{3.13}$$

Pro samotnou detekci, ať využíváme jakoukoli metriku, lze použít lokálního operátoru vektor gradient, jenž porovnává vektor zkoumaného bodu s vektory bodů v osmiokolí zkoumaného bodu. Následným výsledkem tohoto operátoru je maximální hodnota těchto porovnávání, přičemž rovnice při využití Euklidovské vzdálenosti je označena jako 3.14 a při využití úhlu mezi vektory je hodnota operátoru určena rovnicí 3.15.

$$E_{VG} = \max_{i=1\ldots8} (||\vec{v}_i(x,y) - \vec{v}_0(x,y)||) \tag{3.14}$$

$$S_{VG} = \max_{i=1\ldots8} \left(\sqrt{1 - \left(\frac{\vec{v}_i^T(x,y) \cdot \vec{v}_0(x,y)}{||\vec{v}_i(x,y)|| \cdot ||\vec{v}_0(x,y)||}\right)^2}\right) \tag{3.15}$$

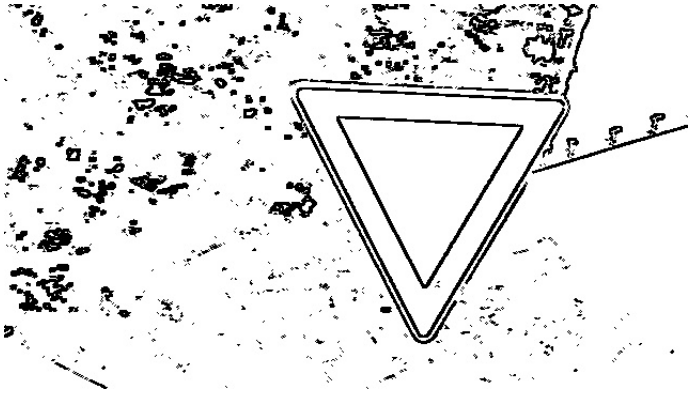
Ve článku [19] je dále popsáno využití kombinace Euklidovské vzdálenosti s úhlem mezi vektory a jak vhodně zvolit váhu jednotlivých metrik v závislosti na daných bodech a jejich barevných vektorech. V článku jsou prezentovány dvě možnosti na jakých hodnotách je možné založit tuto váhu, a to buď na intenzitě dané barvy, nebo na její sytosti. V článku je zvolena sytost barvy a to zejména kvůli vyššímu šumu v barevném tónu než v intenzitě barvy, když je sytost barvy nízká. Pro každý bod je tedy následně nutné spočítat tento parametr určený sigmoidou 3.16, přičemž hodnoty *slope* a $offset$ je nutné zvolit v závislosti na tom, jaké přesné chování od sigmoidy očekáváme, a jejich podoba může být aplikačně závislá. Hodnotu této váhy je ovšem nutné zvolit v závislosti na dvou bodech a v dané publikaci je navrhováno využití geometrického průměru v podobě rovnice 3.17.

$$\alpha(S) = \frac{1}{1 + e^{-slope(S - offset)}} \tag{3.16}$$

$$\rho(S_1, S_2) = \sqrt{\alpha(S_1) \cdot \alpha(S_2)} \tag{3.17}$$

Pokud dáme tedy dohromady získanou váhu mezi Euklidovskou vzdáleností a úhlem mezi vektory a použijeme lokální operátor vektor gradient, tak získáme rovnici prezentovanou jako 3.18. V článku je také prezentováno, že bylo dosaženo lepších výsledků, když nebyla určena absolutní hodnota, nad kterou má být již bod považován za hranu, ale spíš je vhodné využití 15 % bodů s nejvyšší hodnotou jako hrany. Výsledek, který byl takto získán, lze vidět na obrázku 3.4

$$C_{VG} = \max_{i=1\ldots8} \begin{pmatrix} \rho(S_1, S_2)\sqrt{1 - \left(\frac{\vec{v}_i^T(x,y) \cdot \vec{v}_0(x,y)}{||\vec{v}_i(x,y)|| \cdot ||\vec{v}_0(x,y)||}\right)^2} + \\ [1 - \rho(S_1, S_2)] \cdot ||\vec{v}_i(x,y) - \vec{v}_0(x,y)|| \end{pmatrix} \tag{3.18}$$



Obrázek 3.4: Ukázka výsledku detekce hran s využitím celého RGB prostoru pro obrázek 3.3a

3.3 Detekce obrysů

V následující části bude prezentován používaný algoritmus pro detekci obrysů v obraze využívaný v mé práci. Algoritmus je založen na článku Satoshi Suzukiho [15] a také na publikaci Azrieala Rosenfelda [12] představující algoritmus známý jako „sledování pomezí“ (border following).

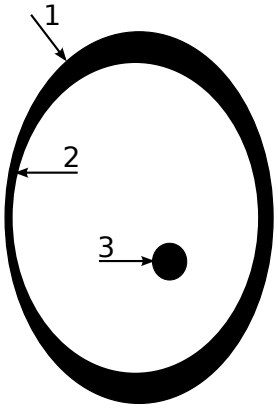
Algoritmus umožňuje v binárním obraze provést detekci a rozponání jednotlivých obrysů, přičemž tento binární obraz, ve kterém budou obrysy detekovány, by měl být získán pomocí detektoru hran. V článku je využíváno následujících konvencí, které budou také v této části také využívány. Souřadnice bodu v obraze jsou určeny v pořadí $(row, column)$ a levý horní roh má souřadnice $(0, 0)$.

V článku je zavedena terminologie nutná pro vysvětlení algoritmu, jenž bude dále vysvětlena. Využívají se tři typy bodů, které mají různé značení.

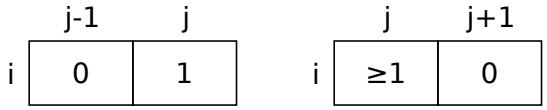
- Body označené 0, které nejsou ani hranou ani označeným obrysem.
- Body označené 1, které jsou považovány za body hrany, které ještě nebyly označeny jako obrys.
- Body označené jinou hodnotou, jenž může být kladná i záporná, označují již detekovaný obrys.

V článku jsou dále také obrysy děleny na dva typy a to vnitřní a vnější. Vnější obrys je definován jako rozmezí mezi komponentou složenou z kladných hodnot binárního obrazu a komponentou složenou z nulových hodnot binárního obrazu, která komponentu z kladných hodnot přímo obklopuje. Vnitřní obrys je definován jako rozmezí mezi komponentou kladných hodnot, která přímo obklopuje komponentu nulových hodnot. Další vysvětlení této definice lze nalézt na obrázku 3.5.

Algoritmus detekce obrysů probíhá následovně. Obraz je procházen po řádcích zleva doprava a procházení je zastaveno, pokud je nalezen bod (i, j), jenž splňuje podmínku pro počáteční bod obrysu a to ať už vnějšího nebo vnitřního. Tyto podmínky jsou popsány na obrázku 3.6. Pokud bod splňuje podmínku pro vnější i pro vnitřní obrys, musí být považován za počáteční bod vnějšího obrysu. Danému obrysu je pak přiřazeno kladné číslo, jenž jej jednodznačně identifikuje, označíme jej jako NBD.



Obrázek 3.5: Jednotlivé typy obrysů. Obrysy 1 a 3 jsou vnějšími obrysy a obrys 2 je vnitřní obrys.



(a) Vnější hrana (b) Vnitřní hrana

Obrázek 3.6: Podmínky pro různé typy hran

Následně jsou všechny body tohoto obrysu procházeny pomocí algoritmu sledování pomezí, jenž je podrobně popsán v článku Azrieala Rosenfelda [12]. Tento způsob lze popsat následovně. Při procházení obrysu je udržováno pomezí dvou komponent na jedné straně a prochází se takovýmto způsobem do té doby než se opět narazí na počáteční bod procházení obrysu. Body obrysu jsou pak následně označovány podle následujících pravidel.

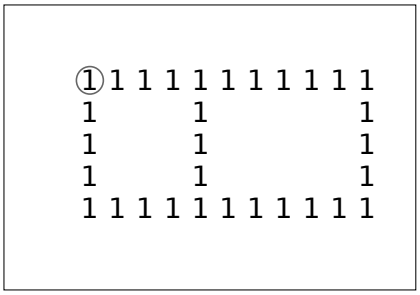
1. Pokud bod $(p, q+1)$ obsahuje nulovou hodnotu, bod (p, q) obsahuje nenulovou hodnotu a zároveň sledované pomezí leží mezi těmito dvěma body, pak je bod obrysu (p, q) označen unikátním číslem -NBD.

2. Jinak je bod obrysu (p, q) označen číslem NBD, ovšem pouze tehdy, pokud již nebyl dříve označen jako obrys.

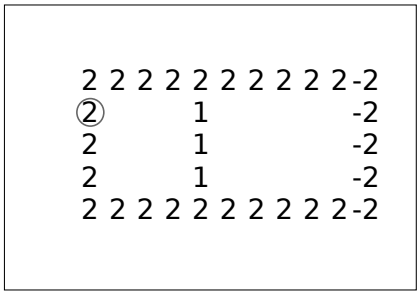
Takto algoritmus probíhá dokud skenování obrazu nedojde do pravého dolního rohu, kdy algoritmus končí se zpracováváním. Celý proces algoritmu je také graficky znázorněn a dále popsán na obrázku 3.7. Následně je nutné získané obrysy již pouze extrahovat z obrazu a zvolit jejich vhodnou reprezentaci. Pro tento účel lze využít ať už Freemanova kódu, jenž je popsán v článku [7], nebo obrys aproximovat pomocí úseček, které budou určeny body. Další možností je reprezentovat daný obrys pomocí všech jeho bodů.

3.4 Detekce geometrických útvarů

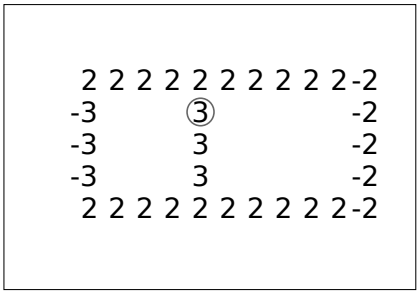
V této kapitole budou popsány dva možné způsoby pro detekci geometrických útvarů v obraze. Myšlenka obou algoritmů bude vysvětlena na případu detekce přímek a následně bude ukázáno jak lze detekovat další geometrické objekty jako například kružnice.



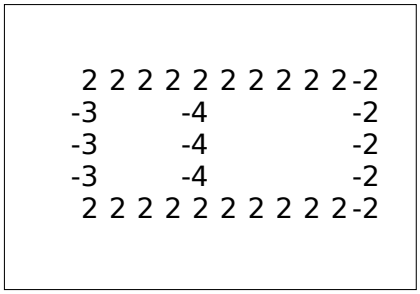
(a) V označeném bodě byla splněna podmínka pro vnější hranu a daný obrys bude označen číslem 2



(b) Označený bod splňuje podmínku pro vnitřní hranu a tento obrys bude označen jako 3



(c) Označený bod také splňuje podmínku definující vnitřní hranu a bude označen číslem 4



(d) Výsledné označení, jelikož už žádný další bod nesplňuje ani jednu z podmínek

Obrázek 3.7: Proces detekce obrysů

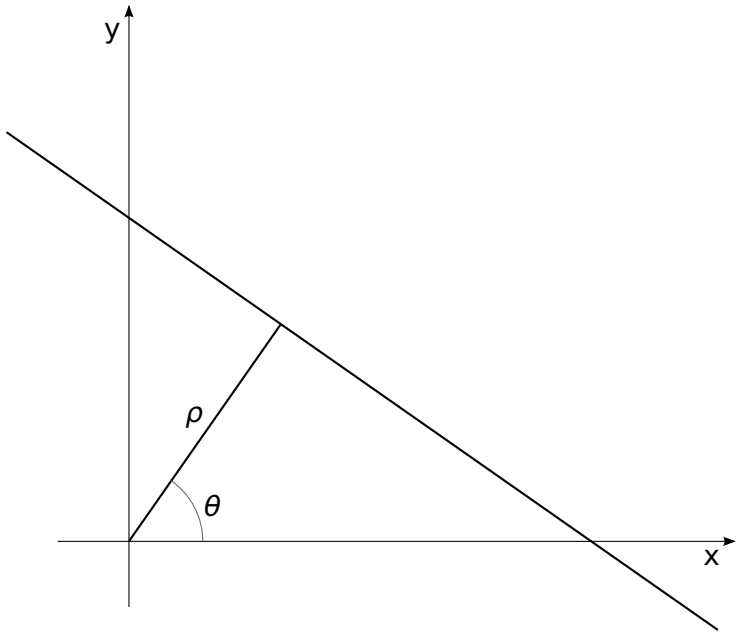
Při psaní této kapitoly jsem vycházel především z článku Richarda Dudy [4] popisující Houghovu transformaci a také článku Martina Fischlera [5], který představuje algoritmus RANSAC pro detekci křivek v obraze.

3.4.1 Houghova transformace

Houghova transformace, která umožňuje detekci přímek, vychází z rovnice přímky 3.19 a jak si lze všimnout, tak daná přímka je definována dvěma parametry θ a ρ. Úhel θ je úhel, který svírá normála přímky procházející počátkem kartézského souřadnicového systému s kladnou částí osy x a ρ je vzdálenost této přímky od počátku. Tyto hodnoty jsou také znázorněny na obrázku 3.8.

$$x \cos\theta + y \sin\theta = \rho \tag{3.19}$$

Jelikož pro účely Houghovy transformace lze umístit počátek souřadnicového systému do levého spodního rohu obrázku, tak můžeme omezit úhel θ na interval $< 0; \pi)$ a následně je pak každá přímka jednoznačně určena těmito parametry θ a ρ, tedy každé kombinaci těchto parametrů odpovídá právě jedna přímka a každé přímce v obraze odpovídá právě jedna kombinace těchto parametrů. Pokud bychom vzali všechny body jedné přímky, přičemž bod má souřadnice (x_i, y_i), a pro každý tento bod určili křivku 3.20, tak zjistíme, že se všechny takto definované křivky protínají v jednom bodě, který odpovídá parametrům dané přímky. Toto lze vidět na obrázku 3.9, kde jsou tyto křivky sestrojené pro body z jedné přímky.



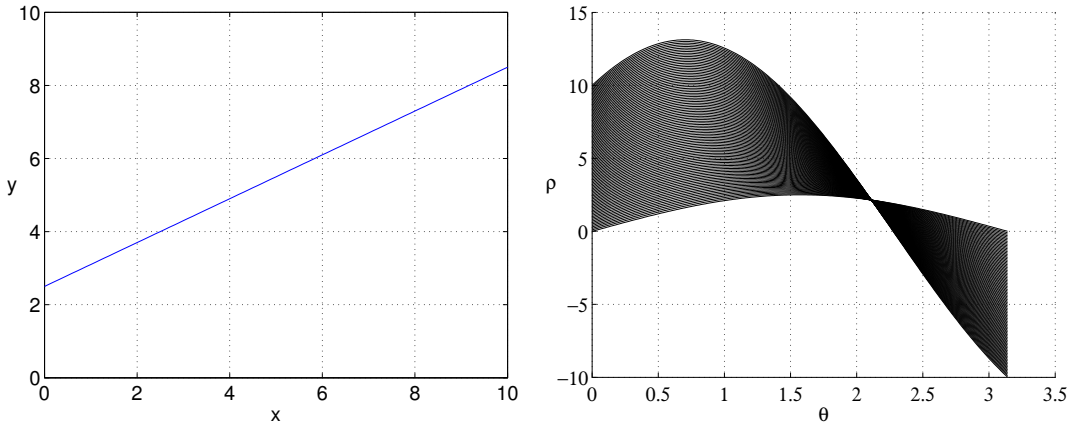
Obrázek 3.8: Význam ρ a θ

$$\rho = x_i \cos\theta + y_i \sin\theta \tag{3.20}$$

Nyní předpokládejme, že máme obraz, ve kterém je přítomna konečná množina bodů $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, jež byly získány například pomocí detektoru hran, a chceme určit, zdali tato množina nebo nějaká její podmnožina bodů tvoří přímku, případně i více

přímek. Pro tuto detekci se využívá právě průsečíků křivek 3.20, které byly získány pro všechny body z této množiny. Pokud bychom ovšem chtěli tyto průsečíky určovat přesně, bylo by to výpočetně náročné, jelikož složitost roste kvadraticky s počtem zkoumaných bodů. Poněvadž velmi často není nutné znát parametry těchto přímek naprosto přesně lze využít následující optimalizaci.

Před začátkem běhu algoritmu si určíme s jakou chybou chceme přímky detekovat a vytvoříme prostor, do kterého budeme zaznamenávat křivky určené pro jednotlivé body obrázku. Zmíněný prostor se většinou implementuje jako dvojrozměrné pole celých čísel, kde každý prvek tohoto pole odpovídá určitému úhlu θ a vzdálenosti ρ. Při vytváření tohoto pole se všechny hodnoty inicializují na hodnotu 0. Poté je pro každý bod v obraze, který je označen jako hrana, určena rovnice křivky 3.20 a následně je hodnota prvku Houghova prostoru, jímž křivka s definovanou chybou prochází, zvýšena o jedna. Pokud toto provedeme pro všechny body, tak získáme pole, kde v každém prvku je číslo, které určuje kolik křivek právě tímto bodem procházelo.



(a) Detekovaná přímka

(b) Příslušný Houghův prostor

Obrázek 3.9: Příklad detekce Houghovou transformací. Detekovaná přímka má rovnici $y = 0,6x + 2,5$. Odpovídající parametry tedy jsou $\rho \doteq 2,14$ a $\theta \doteq 2,11$

Jak bylo výše zmíněno, tak všechny křivky 3.20 pro body jedné přímky se protínají v jednom bodě, následně tedy stačí najít lokální maxima ve vytvořeném prostoru a pro tyto hodnoty zjistit odpovídající hodnoty úhlu θ a vzdálenosti od počátku ρ a získat tak parametry přímek nacházející se v obraze. Případně lze ještě definovat další pravidla pro detekované přímky, jako například minimální počet bodů, které musí danou přímku tvořit. Příklad takto nalezených přímek lze vidět na obrázku 3.10.

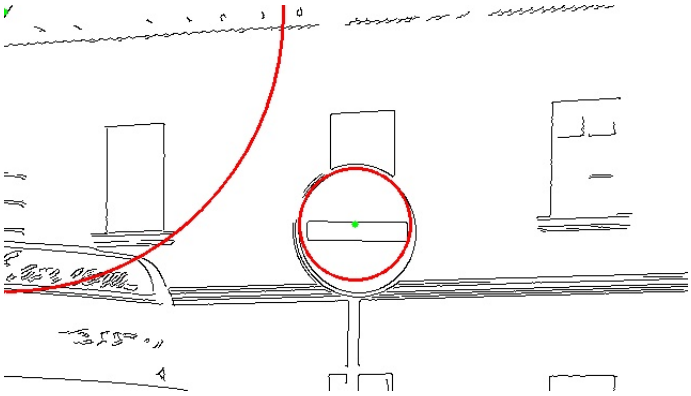
Úprava pro detekování kružnice

Pomocí malých změn v algoritmu lze získat z obrazu rovněž kružnice. Tentokrát se vychází z rovnice 3.21, přičemž střed kružnice je určen hodnotami a a b, parametr c pak následně určuje poloměr dané kružnice.

$$(x - a)^2 + (y - b)^2 = c^2 \tag{3.21}$$

(a) Originální obrázek (b) Vyznačené detekované přímky

Obrázek 3.10: Obrázek detekovaných přímek pomocí Houghovy tranformace



Obrázek 3.11: Detekované kružnice pomocí Houghovy transformace

$$(x_i - a)^2 + (y_i - b)^2 = c^2 \tag{3.22}$$

Většina algoritmu je naprosto obdobná líší se ovšem útvar, do kterého je každý bod transformován. Již to není křivka jako v případě přímek a rovnice 3.20, ale je to kužel, který je pro každý bod (x_i, y_i) určen rovnicí 3.22. Z tohoto důvodu je také nutné vytvořit již trojrozměrné pole, do kterého se budou dané kužely zaznamenávat. Dále již je zbytek algoritmu obdobný.

3.4.2 RANSAC

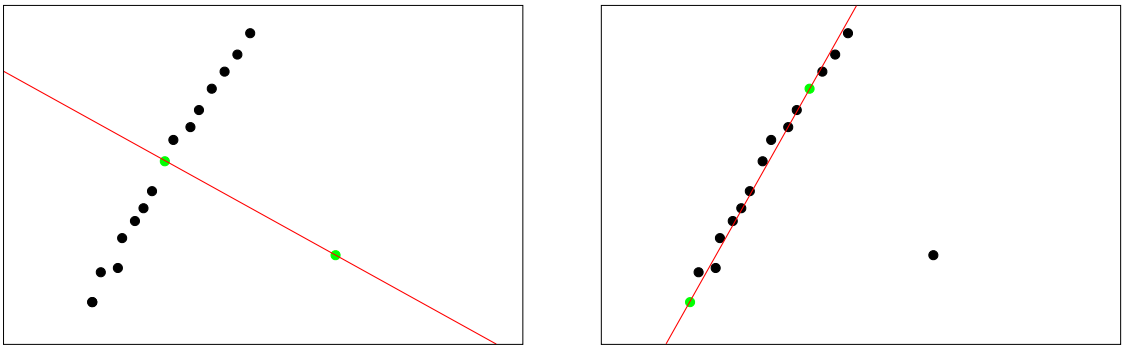
Detekce geometrických útvarů pomocí algoritmu RANSAC je založena na odlišném přístupu než výše zmíněná Houghova transformace. Algoritmus je založen na náhodném výběru dvou bodů a testování hypotézy, že tyto dva body tvoří přímku, přičemž je ověřováno, kolik dalších bodů tuto hypotézu potvrdí.

Pro algoritmus je nutné určit tři parametry, které mají zásadní význam pro získaný výsledek. Jedná se o počet náhodných pokusů k vybrání dvou bodů, maximální vzdálenosti bodu od přímky w, aby tento bod podpořil hypotézu, že daná přímka se v obraze vyskytuje a v neposlední řadě dolní hranice t určující, kolik procent bodů musí podpořit danou hypotézu. Vhodnou volbou těchto parametrů se zabývá článek [5], ve kterém byl tento algoritmus představen.

Algoritmus dále probíhá následovně. Z množiny bodů, pro které chceme provést detekci přímky, jsou náhodně vybrány dva body a následně je určena přímka, kterou tyto body jednoznačně určují a definujeme hypotézu, že v obraze je obsažena přímka definovaná právě těmito dvěma body. Následně je pro všechny další body určena vzdálenost od této přímky a testuje se, jestli tato vzdálenost je menší než maximální vzdálenost w. V kladném případě se tento bod označí jako bod podporující hypotézu a poté co jsou takto otestovány všechny body zjistí se, jestli alespoň t procent bodů tuto hypotézu podpořilo. Pokud je i toto splněno, tak je původní hypotéza, tedy že náhodně vybrané body tvoří přímku, prohlášena za správnou a všechny body, které tuto hypotézu podpořily se považují za body přímky.

Na obrázku 3.12 lze vidět dva případy náhodné volby bodů pro určení přímky, v případě 3.12a se hypotéza nepotrvdila, ovšem když se změnil výběr bodů, které definují přímku, tak jako na obrázku 3.12b, tak se již hypotéza potvrdila.

Při detekování kružnic je nutné udělat pouze změnu ve výběru bodů. Jelikož je každá kružnice jednoznačně určena třemi body, tak je nutné náhodně vybrat právě tři body pro její určení. Následně je obdobně testována vzdálenost dalších bodů od této kružnice.



(a) Při náhodném zvolení bodů, jež je naznačeno na tomto obrázku nebylo nalezeno dostatečné množství dalších bodů, které by tuto hypotézu potvrdily

(b) V tomto případě již byl nalezen dostatečný počet bodů, které potvrzují hypotézu přítomnosti přímky určené danými body

Obrázek 3.12: Detekce přímek pomocí algoritmu RANSAC. Zeleně jsou označené náhodně vybrané body, které definují přímku, pro kterou je ověřována hypotéza, že je přítomna v obraze.

Kapitola 4

Detekce dopravních značek založená na tvarech

Já jsem svoji práci založil na algoritmu publikovaném v článku [8], jenž byl popsán v kapitole 2.3. Algoritmus využívá pro detekci dopravních značek jejich geometrických vlastností. Tento algoritmus jsem zvolil zejména proto, že jsem jej považoval za robustnější, jelikož tvar dopravních značek je ve většině států konzistentní a mění se případně pouze barva značek. Tento způsob detekce dopravních značek také sliboval velmi dobré výsledky, jichž bylo v rámci článku [8] dosaženo.

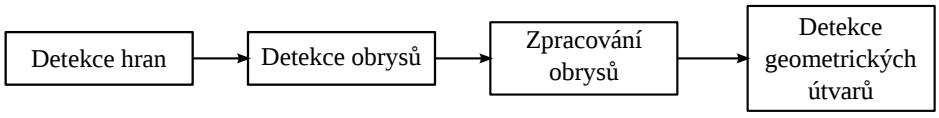
V další části této kapitoly budou popsány zejména úpravy, které jsem prováděl ke zlepšení výsledků, kterých jsem dosahoval pomocí tohoto algoritmu.

4.1 Detekce hran

V rámci práce jsem nejprve pracoval s Cannyho detektorem hran, tak jak byl popsán v kapitole 3.2.1. Ovšem v průběhu času se ukázalo, že tento detektor hran nepostačuje v případech, kdy přechod mezi značkou a okolím byl v šedotónovém obrázku příliš nevýrazný a nebyla tedy hrana detekována. Z tohoto důvodu jsem hledal způsoby, jak vylepšit detekci hran, aby k těmto problémům nedocházelo nebo byly alespoň v co nejvyšší míře eliminovány.

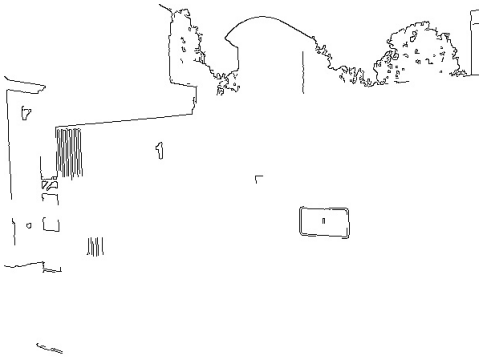
Nejprve jsem tedy prováděl testování se stejným algoritmem Cannyho detektoru hran, jemuž byl předkládán obraz v odstínech šedi a snažil jsem se upravit prahy, které jsou využívány pro prahování s hysterzí. Tento způsob vedl k detekci více hran dopravních značek nacházejících se ve stínech a podobně špatných světelných podmínkách. Ovšem jelikož tyto prahy byly již příliš nízké, bylo detekováno také velké množství šumu.

Z tohoto důvodu jsem otestoval detektor hran využívající celého RGB barevného prostoru, který využívá pro určení rozdílnosti mezi dvěma body Euklidovskou vzdálenost i úhel mezi vektory, jež jsou tvořeny jednotlivými barevnými složkami bodu, přičemž algoritmus byl popsán v kapitole 3.2.2. Tento algoritmus produkoval již dostatečně dobré výsledky,

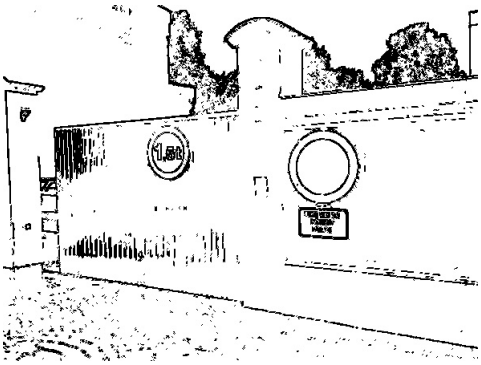


Obrázek 4.1: Proces detekce dopravních značek

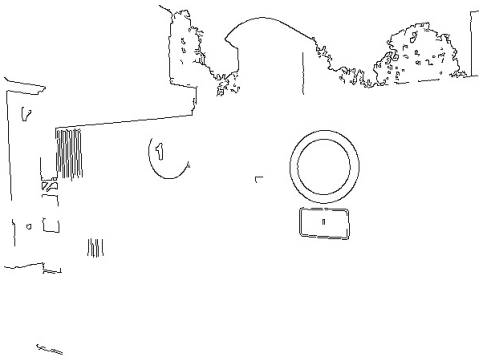
(a) Originální obrázek



(b) Cannyho detektor hran, šedotónový obrázek



(c) Detekce hran pomocí algoritmu využívající celý RGB barevný prostor



(d) Cannyho detektor hran, průměr zelené a modré barvy

Obrázek 4.2: Porovnání jednotlivých detektorů hran. Cannyho hranový detektor má v obou případech shodné nastavení prahů a to 100 a 200.

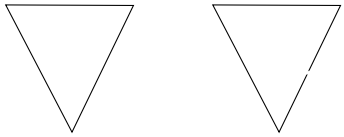
ovšem ukázalo se, že je příliš pomalý pro využití v mé práci, která si klade za cíl detekovat a klasifikovat dopravní značky v reálném čase.

Následně jsem otestoval Cannyho detektor hran, kterému ovšem není předáván obraz, jenž byl získán pomocí rovnice 3.1 pro převod obrazu v RGB barevném prostoru do odstínu šedi. Místo toho je detektoru hran předán obraz, jenž byl získán jako aritmetický průměr zelené a modré složky obrazu. Tímto se eliminuje červená složka a jsou více viditelné přechody mezi okolím a dopravní značkou. Tento způsob relativně dobře funguje i pro modré a žluté značky.

Experimentoval jsem také s dalšími barevnými kombinacemi, například jsem zkoušel detekovat hrany v obrazu, který obsahuje pouze barevný odstín modelu HSV. Tyto další experimenty již ovšem nepřinesly další zlepšení algoritmu. Srovnání výše uvedených způsobů detekce hran lze vidět na obrázku 4.2.

4.2 Zpracování obrysů

Po detekci hran jsou detekovány obrysy a tyto obrysy jsou dále zpracovávány. To, jakým způsobem toto zpracování probíhá, bude následně popsáno. Cílem těchto operací s obrysy objektů v obraze je především vyloučit z dalšího zpracování takové obrysy, které jistě ne-



(a) (b)

Obrázek 4.3: Obrys označený 4.3a má nenulový obsah, naproti tomu obrys označený 4.3b má nulový obsah kvůli nepatrné nespojitosti objektu.

mohou být dopravní značkou.

Operace se dělí na dva typy. Jeden typ pracuje přímo s obrysem a pokud daný obrys projde kontrolou těchto pravidel, je vypočítán obdélník, který jej ohraničuje. Obdélník je určován takový, aby měl strany rovnoběžné se souřadným systémem. Druhý typ těchto operací následně pracuje již s tímto ohraničujícím obdélníkem.

Vyhodnocení obrysu

Pro zpracování obrysu jsou použité dva testy, které mají rozhodnout, jestli obrys může být obrysem dopravní značky.

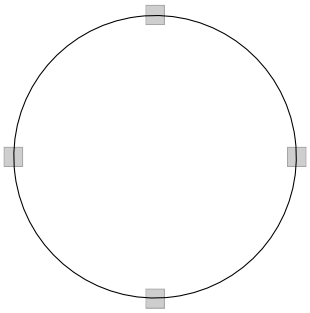
Nejprve je určen obsah plochy, kterou obrys ohraničuje. Pokud je tento obsah větší než minimální požadovaný obsah, je obrys rovnou považován za potenciální dopravní značku a předán dále ke kontrolám. Pokud obrys tomuto pravidlu nevyhoví, tak ještě může být prohlášen za možný výskyt dopravní značky. Tedy dá se toto považovat za podmínku dostačující, ale nikoli nutnou.

K tomuto přístupu jsem byl nucen přistoupit z důvodu, že ne vždy jsou hrany obrysu detekovány naprosto dokonale a může se vyskytnout obrys dopravní značky, který není uzavřený a má tedy nulový obsah. Tento problém je znázorněn na obrázku 4.3.

Další kontrolou, která je prováděna při zpracování obrysu, je ověření barev v okolí obrysu. Tato kontrola je založena na myšlence, že by obrys dopravní značky měl mít ve svém okolí červenou nebo modrou barvu. Kontrola se neprovádí u všech bodů obrysu, ale pouze na vzorku, přičemž tyto vzorky jsou rovnoměrně rozloženy v obryse, aby tato kontrola měla dobrou vypovídací hodnotu.

Pro každý bod je vyhodnoceno okolí, které sahá 2 body na každou stranu, je tedy celkem vyhodnoceno 25 bodů. Pro každý bod je pak brána maximální dosažená hodnota ohodnocení těchto okolních bodů. Tedy pro bod platí rovnice 4.2, přičemž hodnoty x' a y' jsou omezeny pomocí podmínek 4.3.

Ohodnocení barvy daného bodu o souřadnicích (x, y) jsem experimentálně otestoval a nejlepších výsledků jsem dosáhl pomocí rovnice 4.1, přičemž H značí barevný odstín, jenž má daný bod. Daný obrys splní tuto podmínku pokud průměr barevných ohodnocení $E(x, y)$ pro všechny testované body dosáhne alespoň předem stanovené procentuální hranice. Ve své práci tuto hranici mám stanovou jako 50 %. Rovnice jsou koncipovány pro hodnotu H, která nabývá celočíselné hodnoty v intervalu $< 0; 180 >$.



Obrázek 4.4: Hledání barev dopravních značek v okolí obrysu. Pro jednoduchost jsou znázorněny pouze 4 testovací body, v reálné aplikaci je těchto bodů ovšem 32. V každém šedém obdélníku, který má velikost 5 x 5 bodů, je nalezen bod, který má maximální hodnotu určenou rovnicí 4.1 a následně jsou tyto body průměrovány a porovnávány s prahem.

$$f(x,y) = \begin{cases} 1 & H < 40 \\ 1 & H > 140 \\ \frac{80-H}{40} & H < 80 \\ \frac{H-120}{40} & H > 120 \\ 0 & \text{jinak} \end{cases} \tag{4.1}$$

$$E(x,y) = \arg\max_{x',y'} f(x',y') \tag{4.2}$$

$$\begin{aligned} (x-2) \leq x' \leq (x+2) \\ (x-2) \leq x' \leq (x+2) \end{aligned} \tag{4.3}$$

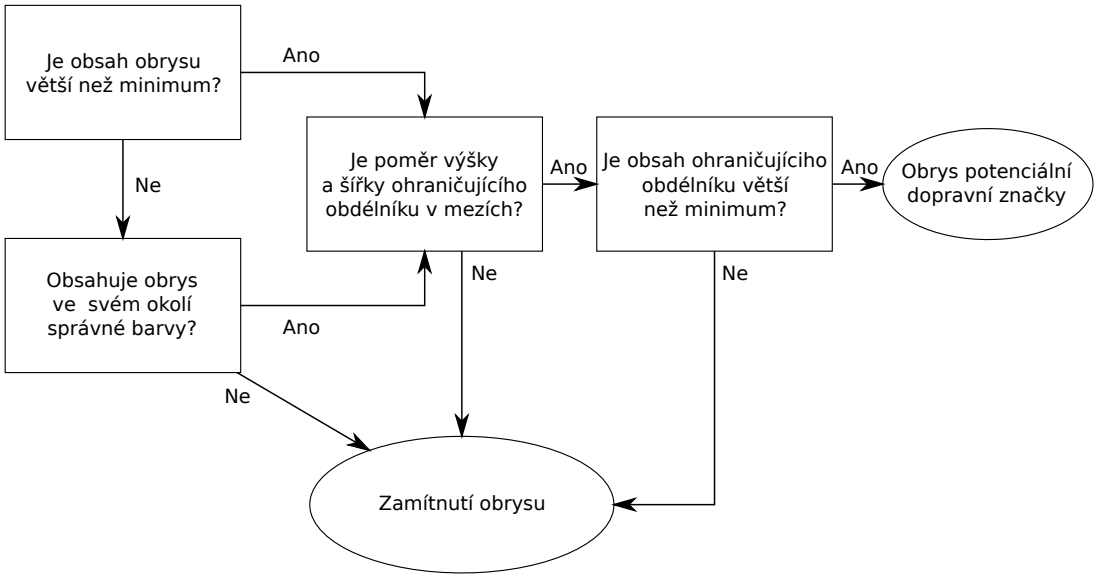
Vyhodnocení ohraničujícího obdélníku

Pro ohraničující obdélník, který byl spočítán pro daný obrys, musí platit také dvě pravidla. Tato pravidla si kladou za cíl eliminovat obrysy, které již prošly podmínkami pro obrys, ale stejně by neměly být považovány za potenciální dopravní značku.

První takovouto podmínkou je kontrola, jestli nemá obdélník příliš rozdílnou šířku a výšku, protože dopravní značky mívají tento ohraničující obdélník téměř ve tvaru čtverce. Pokud by byla detekována dopravní značka za jízdy, která by měla ohraničující obdélník příliš vzdálený čtverci, tak se bude velmi pravděpodobně jednat o dopravní značku nacházející se na jiné dopravní komunikaci než je aktuálně vozidlo. Toto není žádoucí, protože by to mohlo velmi zmást případný systém, který využívá detekci dopravních značek pro řízení automobilu. Z tohoto důvodu musí platit pro obdélník nerovnice 4.4, přičemž w značí šířku obdélníku a h jeho výšku.

$$0,65 < \frac{w}{h} < 1,35 \tag{4.4}$$

Další a již konečnou podmínkou je kontrola obsahu ohraničujícího obdélníku, kterou již musí obdélník splnit a není tomu jako u kontroly obsahu obrysu. Tato podmínka byla zařazena především kvůli malým obrysům, které splnily test na minimální procento správně barevných bodů v okolí obrysu, ale jsou zároveň také příliš malé na to být dopravní značkou.



Obrázek 4.5: Výsledné schéma vyhodnocování obrysu

4.3 Geometrické objekty

Po zpracování těchto obrysů a jejich ohraničujících obdélníků jsou v každém obdélníku, jenž je označen jako potenciální dopravní značka, hledány geometrická primitiva jako kružnice a přímky. Algoritmus hledání těchto geometrických útvarů bude diskutován v kapitole 5 zabývající se implementací mého systému pro detekci dopravních značek.

V tomto kroku je v daném obdélníku hledána kružnice, jejíž poloměr r splňuje podmínku označenou jako 4.7. Hodnoty b a s jsou určeny rovnicemi 4.5 a 4.6, přičemž hodnoty w a h jsou šířkou a výškou obdélníku ve kterém je kružnice vyhledávána. Pokud je takováto kružnice nalezena, je obdélník s konečnou platností prohlášen za potenciální výskyt dopravní značky.

$$b = \begin{cases} h & h > w \\ w & \text{jinak} \end{cases} \tag{4.5}$$

$$s = \begin{cases} h & h < w \\ w & \text{jinak} \end{cases} \tag{4.6}$$

$$\frac{s}{2} - 15 < r < \frac{b}{2} + 15 \tag{4.7}$$

Pokud není v daném výřezu nalezena kruhová značka, jsou hledány ostatní typy dopravních značek a to zejména čtvercové a trojúhelníkové. Pro oba tyto typy platí, že nejprve jsou detekovány přímky v daném výřezu a pak je proveden test, jestli z těchto přímek lze sestrojit rovnostranný trojúhelník nebo čtverec.

Kontrola, zda lze z přímek sestrojit dohromady dané geometrické útvary, je prováděna pouze pomocí úhlů. Pro trojúhelník jsou tedy vyhledávány ke každé přímce alespoň dvě další přímky, které s ní svírají úhel v intervalu $(50°, 70°)$. Pro čtverce je kontrola naprosto obdobná, přímky jsou testovány ovšem na úhel v intervalu $(80°, 100°)$. Test pro čtverce tedy nekontroluje přítomnost poslední čtvrté strany, ovšem ukázalo se, že tato kontrola není potřebná a není příliš zvýšen počet falešných detekcí dopravních značek.

(a) Ohraničující obdélník je spočítán pro nalezenou kružnici

(b) Ohraničující obdélník je spočítán pro obrys

Obrázek 4.6: Rozdíl mezi lokalizací pomocí ohraničujícího obdélníku obrysu a nalezené kružnice

4.4 Výsledná oblast považovaná za dopravní značku

Poslední otázkou, na kterou bylo v průběhu detekce dopravních značek nutné nalézt odpověď, bylo, jak přesně nálézt výsledný obdélník osahující dopravní značku, tak aby tento obdélník tuto značku co nejpřesněji označoval.

Naskýtaly se dvě možnosti jak se k tomuto problému postavit. Je možné označit původní obdélník, který byl určen jako ohraničení obrysu, nebo lze také u kruhových značek označit jako dopravní značku obdélník, jenž ohraničuje nalezenou kružnici.

Testoval jsem oba dva přístupy a během testování se ukázalo, že detekované kružnice jsou velmi často detekovány nepřesně a pozice reálné značky je často posunutá oproti označenému obdélníku, který určuje potenciální výskyt dopravní značky. Rozdíly lze nalézt na obrázku 4.6.

Z tohoto důvodu jsem zvolil jako dopravní značku původní obdélník, který ohraničoval obrys a ukázalo se, že takovéto označení je velmi přesné a pozice označeného potenciálního výskytu dopravní značky se velmi často shoduje s reálnou pozicí dopravní značky.

Kapitola 5

Implementace

Tato kapitole popisuje praktickou implementaci systému pro detekci dopravních značek, jenž byl popsán v kapitole 4. Program umožňuje nejen detekci dopravních značek, ale také jejich klasifikaci.

5.1 Použité nástroje

Pro implementaci většiny algoritmů, které zpracovávají obraz, jsem využil knihovny OpenCV, která se zaměřuje na počítačové vidění a obsahuje velké množství funkcí a algoritmů, které lze využít právě pro počítačové vidění.

Pro klasifikaci dopravních značek jsem použil program Tomáše Svobody, který byl implementován v rámci jeho diplomové práce [16]. Z tohoto programu nebyly využity veškeré jeho funkce, ale pouze ty, které provádějí klasifikaci dopravních značek. Jelikož tento zmíněný program využívá pro svojí funkci dalších knihoven, byl jsem nucen je také zařadit do mé práce. Jedná se především o knihovny TinyXML a SVMLight.

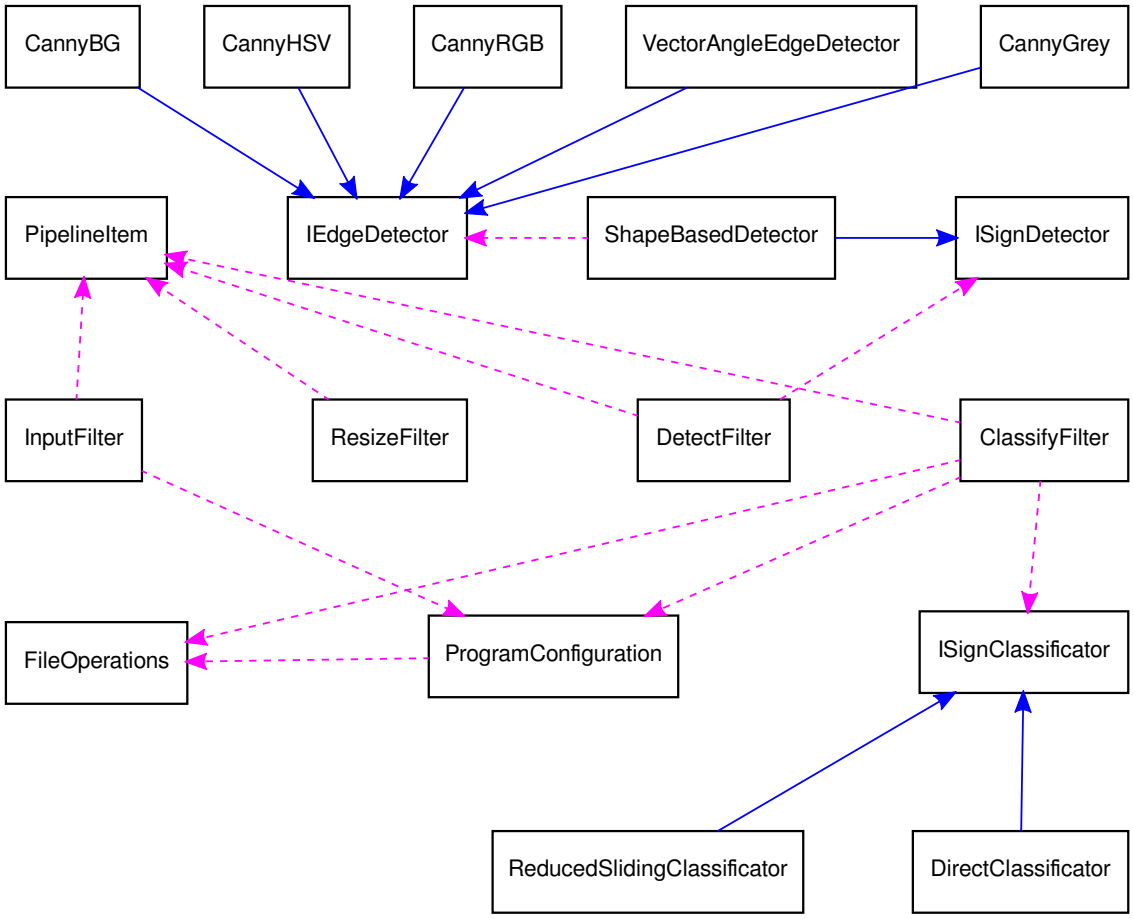
Pro samotné naprogramování mé práce jsem zvolil jazyk C++ a to především pro jeho kombinaci objektového přístupu a také rychlosti. Podpůrné vyhodnocovací nástroje jsem implementoval ve skriptovacích jazycích Python a Bash, především pro jednoduchost vývoje v těchto jazycích.

5.2 Implementace programu pro detekce dopravních značek

Program implementovaný v rámci mé bakalářské práce umožňuje načítání obrázků a videí, následnou detekci a klasifikaci dopravních značek v těchto obrázcích. Informace o detekovaných dopravních značkách jsou vypisovány do XML souboru. Dále je také možné tyto značky vyznačit do obrazu a tento obraz uložit, aby bylo možné zkontrolovat správnou polohu dopravní značky, případně vytvořit video s detekovanými dopravními značkami.

5.2.1 Implementace detekce dopravních značek

Pro implementaci detekce dopravních značek, která je popsána v kapitole 4, jsem vytvořil třídu `ShapeBasedDetector`, která je potomkem abstraktní třídy `ISignDetector`. Pro určení jaký způsob detekce hran má být využit, jsem zvolil využití návrhového vzoru „vložení závislosti“ (Dependency Injection), tedy třídě `ShapeBasedDetector` je vždy v konstruktoru předán ukazatel na třídu, která provádí detekci hran. Tato třída musí být po-



Obrázek 5.1: Diagram spolupráce jednotlivých tříd. Fialová úsečka značí vztah používání a modrá dědičnost.

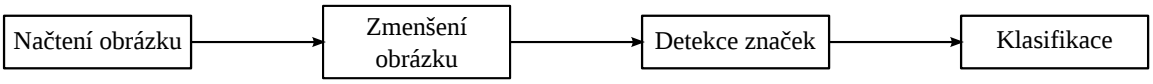
tomkem abstraktní třídy `IEdgeDetector`. Všechny třídy lze vidět na obrázku 5.1, který obsahuje diagram spolupráce jednotlivých tříd.

Dále jsem také testoval, jaký algoritmus pro detekci kružnic a přímek využít v mé práci. Zvažoval jsem využití Houghovy trasformace, která byla popsána v kapitole 3.4.1 a algoritmu RANSAC, kapitola 3.4.2. Ovšem mnou použitá implementace algoritmu RANSAC [11] se ukázala jako příliš pomalá pro využití v mé práci a z tohoto důvodu využívám Houghovy transformace.

5.2.2 Celková koncepce programu

Celý program je koncipován jako řetězec filtrů, které zpracovávají a transformují vstupní obraz. Tyto filtry si předávájí ukazatel na strukturu, která nese veškeré potřebné informace o daném obraze. Tento způsob byl zvolen z důvodu, že jsem dříve experimentoval se zpracováním ve více vláknech, přičemž každý takovýto filtr byl rozdělen do jednoho vlákna. Funkčnost filtrů je následující a celkové zařazení filtrů lze vidět také na obrázku 5.2.

1. První filtr se stará o načítání obrázků a videí a přiřazuje jim jednoznačný identifikátor, který se skládá z názvu souboru a u videí se vkládá také číslo snímku v kontextu daného videa.



Obrázek 5.2: Řetězec filtrů

2. Další filtr pouze zmenší obraz na danou velikost a uloží informaci o zmenšení, jaké bylo nutné provést, aby se dala následně rekonstruovat poloha dopravní značky vzhledem k obrazu v původní velikosti.

3. Následující filtr provede detekci dopravních značek v obraze a uloží všechny obdélníky, které vyhodnotí jako potenciální dopravní značku.

4. Poslední filtr má na starosti detekci dopravních značek a následné výstupní informace. Může například uložit výstupní obrázek nebo vypsat informace o detekovaných či klasifikovaných dopravních značkách a to na standardní výstup nebo do souboru, který byl specifikován parametrem při spuštění programu.

5.3 Podpůrné nástroje

V rámci mé práce bylo nutné implementovat také nástroje, které neměly za cíl detekovat dopravní značky, ale pouze nějakým způsobem pomáhat například ve zpracování a vyhodnocování algoritmu. Z tohoto důvodu vznikly nástroje, které mají zejména dva různé účely a to anotaci dat a vyhodnocování výsledků detekce dopravních značek.

Pro anotaci dat jsem implementoval program, který načítá obrázky z dané složky a zobrazuje je na obrazovku. Následným klikáním myší na obraz lze definovat rohy obdélníka, který ohraničuje dopravní značky přítomné v obraze.

Dále jsem také naprogramoval nástroje, které vyhodnotí XML soubor vygenerovaný detektorem dopravních značek porovnáním s referenčním XML souborem. Takovéto skripty jsou dva, přičemž jeden umožňuje vyhodnocení výstupu porovnáním s již anotovanými obrázky. Toto vyhodnocování bude více popsáno v kapitole 6.1. Druhý skript pro vyhodnocení provádí porovnání s ručně anotovanými soubory a určí metriku F-Measure, jež bude popsána v kapitole 6.2.

Kapitola 6

Vyhodnocení

Tato kapitola se zabývá zejména vyhodnocením systému pro detekci a klasifikaci dopravních značek, který byl popsán a diskutován v kapitole 4 a následně popsány detaily jeho implementace v kapitole 5. Při vyhodnocování jsem se zaměřoval zejména na rychlost zpracování a úspěšnost detekce a klasifikace dopravních značek.

Vzhledem k tomu, že jsem pro klasifikaci dopravních značek využil již implementovaný systém, tak jsem vyhodnocoval zvlášť detekci, která byla předmětem mé bakalářské práce.

Vyhodnocení přesnosti a správnosti detekce dopravních značek jsem prováděl nejprve pomocí velkého množství automaticky anotovaných souborů a následně také metrikou F-Measure. Tato dvě vyhodnocení jsou popsána v kapitolách 6.1 a 6.2. Způsob vyhodnocení rychlosti zpracování videí s dopravními značkami je popsán v kapitole 6.3.

6.1 Porovnání výsledků s anotovanými soubory

V první části vyhodnocování výsledků jsem zaměřil na porovnávání dopravních značek, které byly anotovány s využitím programu Tomáše Svobodu v rámci jeho diplomové práce [16], s výsledky, jichž jsem dosahoval pomocí mého způsobu detekce dopravních značek.

Takto anotovaných dopravních značek bylo velké množství a bohužel nebyla často přesně určena jejich poloha. Z tohoto důvodu jsem byl nucen zavést při vyhodnocení určitou toleranci, která ale ani tak nepokryla všechny případy a některé detekované dopravní značky byly vyhodnoceny jako nenalezené. Takovýchto dopravních značek, ale nebylo příliš velké množství, aby to statisticky významně zlepšilo dosažené výsledky.

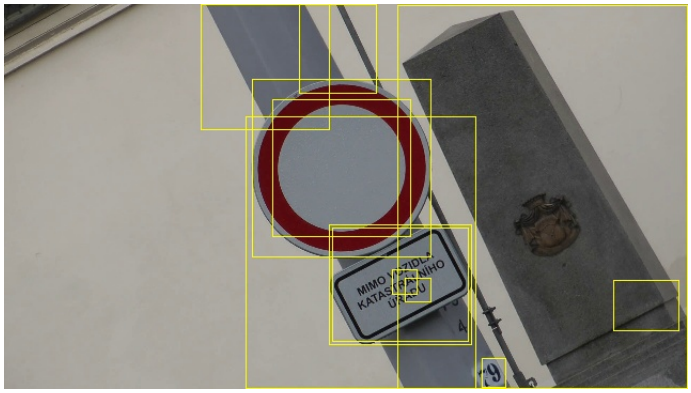
Pro vyhodnocení jsem využil skriptu, který byl popsán v rámce kapitoly 5.3 a dosažené výsledky lze vidět v tabulce 6.1.

Relativně horší výsledky u detekce u dopravní značky „Zakázaný směr jízdy“ jsou způsobeny zejména datovou sadou obsahující tuto dopravní značku. Tato datová sada není příliš velká a obsahuje relativně velké množství dopravních značek, které jsou buď z velké části překryty jiným objektem nebo jsou přítomny v příliš špatných světelných podmínkách.

Jak si lze všimnout v uvedené tabulce, tak pokud byly dopravní značky i klasifikovány, tak bylo dosahováno řádově horších výsledků než při pouhé detekci dopravních značek. Toto je zejména způsobeno mým využitím klasifikátoru dopravních značek, který byl implementován Tomášem Svobodou. Klasifikátor očekával, že bude využit ve spojení s posuvným oknem a dostane tedy více možností klasifikace dopravní značky.

			Miss rate		
Značka	Počet	False alarm	Klasifikace 1	Klasifikace 2	Detekce
Dej přednost	5352	168 226	46 %	33 %	3 %
Zakazáný směr jízdy	1456	41 366	47 %	37 %	14 %
Maximální rychlost	4910	63 407	72 %	34 %	6 %
Stop	1276	34 395	34 %	17 %	5 %
Zákaz vjezdu	1195	24 003	15 %	14 %	4 %
		Průměr	**51 %**	**31 %**	**6 %**

Tabulka 6.1: Výsledky dosažené úspěšnosti detekce. Jako Klasifikace 1 je označano využití klasifikátoru, který přímo zmenší oblast na velikost okna, které se vyhledává. Jako Klasifikace 2 je označeno využití posuvného okna, které ovšem netestuje všechny pozice, ale pouze pár vybraných.



(a) Detekované potenciální dopravní značky

(b) Klasifikované dopravní značky

Obrázek 6.1: Obrázek výsledků klasifikace a detekce dopravních značek

Já tento klasifikátor využívám ovšem mírně odlišným způsobem. Jelikož u dopravních značek, které byly nalezeny pomocí mého systému pro detekci dopravních značek, je velmi přesně označena jejich poloha, tak předkládám klasifikátoru pro každou potenciální dopravní značku pouze jeden výřez a již nevyužívám posuvného okna. Ve druhém způsobu využívám posuvné okno, které netestuje ovšem všechny možné pozice ve výřezu, ale vždy pouze 5 v dané velikosti. Problém s relativně nízkou úspěšností klasifikace by šel z velké míry odstranit přetrénováním modelů, kterých je využíváno pro klasifikaci dopravních značek.

Způsob detekce dopravních značek představený v kapitole 4 je relativně hodně náchylný na falešné detekce dopravních značek, jak lze vidět na obrázku 6.1 a také v tabulce 6.1. Tento problém je ovšem z velké míry kompenzován velmi přesnou lokalizací dopravních značek v obraze. Detektor by měl být využíván pouze jako předstupeň klasifikace, což umožní rychlejší následnou klasifikaci, jelikož tento klasifikátor bude mít velmi přesnou informaci, kde by se případná dopravní značka nacházela.

6.2 F-Measure

V následující části bude popsáno vyhodnocení výsledků mé práce pomocí metriky F-Measure. Nejprve bude metrika F-Measure popsána a ukázano, jak se určuje její hodnota. Následně bude prezentováno samotné vyhodnocení.

Popis F-Measure

Statistický test přesnosti známý jako F-Measure, nebo také jako F-Score, je využíván jako metrika určení, jak přesně byly získány potřebné informace a lze jej využít také pro měření úspěšnosti a přesnosti detekce objektů v obraze. Informace o této metrice jsem čerpal z knihy [10].

$$p = \frac{TP}{TP + FP} \tag{6.1}$$

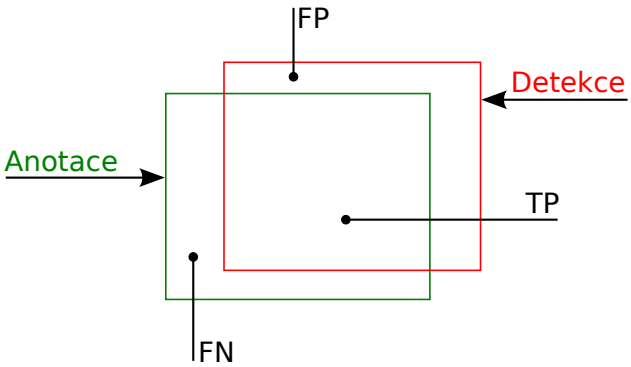
$$r = \frac{TP}{TP + FN} \tag{6.2}$$

$$F = 2 \cdot \frac{p \cdot r}{p + r} \tag{6.3}$$

Rovnice určující výslednou hodnotu F této metriky jsou označeny jako 6.1, 6.2 a výsledně 6.3. Významy hodnot TP, FP a FN lze v kontextu detekce objektů vidět na obrázku 6.2, přičemž následně vypočtená hodnota p určuje kolik procent bodů označených detektorem jako hledaný objekt je opravdu tímto objektem. Hodnota r naopak určuje kolik procent bodů hledaného objektu bylo doopravdy detektorem označeno jako hledaný objekt v obraze.

Vyhodnocení pomocí metriky F-Measure

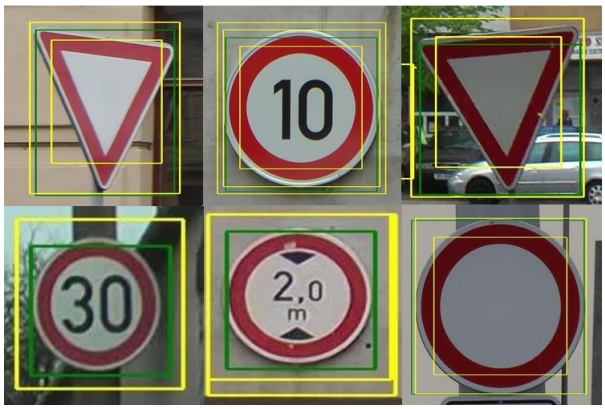
Pro vyhodnocení metriky jsem označil všechny dopravní značky na 100 obrázcích pomocí nástroje pro anotaci dat, který byl popsán v kaptirole 5.3. Takto anotované soubory jsem považoval za refereční pro zpracování metrikou F-Measure. Celkem takto bylo označeno 126 dopravních značek všech typů.



Obrázek 6.2: Význam hodnot TP, FP a FN v kontextu detekce objektů v obraze.

Toto vyhodnocování jsem prováděl zejména z důvodu otestování přesnosti lokalizace detekovaných dopravních značek.

Vyhodnocování jsem prováděl pomocí skriptu, jenž byl popsán v kapitole 5.3. Testována byla detekce a klasifikace zvlášť, přičemž výsledky lze nalézt v tabulce 6.2.



Obrázek 6.3: Ukázka detekovaných dopravních značek a porovnání s anotací používanou pro vyhodnocení metriky F-Measure. Zeleně je označena anotace, žlutě detekované potenciální dopravní značky.

Relativně špatné výsledky této metriky pro značky které byly také následně klasifikovány jsou způsobeny především dvěma faktory. Jelikož vyhodnocení této metriky bylo cíleno především na přesnost detekce dopravních značek, tak byly označovány naprosto všechny značky, které se na obrázcích vyskytly, tedy i ty dopravní značky pro které nebyl natrénován klasifikační model a z tohoto důvodu některé dopravní značky nebyly klasifikátorem označeny. Dále se také v jisté míře přenesl problém s klasifikací, který byl popsán v kapitole 6.1 zabývající se porovnáním výsledků s velkým množstvím automaticky anotovaných dopravních značek.

6.3 Rychlost

Jelikož má práce byla zaměřena na rychlou detekci dopravních značek v obraze, tak bylo nutné určit, jak rychle jsou data zpracovávána a otestovat, jestli by mohla být zpracovávána v reálném čase.

Zpracování značek	F-Measure
Detekce	80,50 %
Klasifikace 1	50,49 %
Klasifikace 2	57,44 %

Tabulka 6.2: Dosažené výsledky metriky F-Measure

Zpracování značek	FPS
Detekce	66
Klasifikace 1	53
Klasifikace 2	30

Tabulka 6.3: Výsledky dosažené rychlosti. Jako Klasifikace 1 je označano využití klasifikátoru, který přímo zmenší oblast na velikost okna, které se vyhledává. Jako Klasifikace 2 je označeno využití posuvného okna, které ovšem netestuje všechny pozice, ale pouze pár vybraných.

Pro otestování jsem zvolil 109 videí, které měly celkem přes 14 hodin. Naměřené výsledky lze vidět v tabulce 6.3.

Měření rychlosti bylo prováděno na počítači s procesorem AMD Phenom X4 945, jenž má frekvenci 3 GHz a 4 GB operační paměti. Testování bylo prováděno na operačním systému Linux a v průběhu měření nebyly pouštěny žádné další výpočetně náročné operace.

Jelikož se v průběhu implementace ukázalo, že formát načítaných dat hraje v rychlosti zpracování velmi významnou roli, tak jsem všechna videa uložil v rozlišení 640×480, jelikož do tohoto rozlišení jsou videa následně v programu zmenšována. Videa byla kódována pomocí kodeku DivX s datovým tokem přibližně 820 kb za vteřinu.

Problém s načítáním dat z pevného disku se v ještě větší míře vysktytuje u zpracování obrázků. Přestože obrázky ve formátu JPEG byly zmenšeny na velikost 640×480, tak jsem dosahoval průměrné rychlosti kolem 18 snímků za vteřinu.

6.4 Zhodnocení výsledků

Jak bylo ukázáno, tak v rámci mé práci byl implementován systém pro detekci dopravních značek, přičemž vstupní data jsou zpracovávána v reálném čase s relativně velkou rezervou, uvažujeme-li za minimum pro zpracování v reálném čase 25 snímků za vteřinu, tak je toto minimum splněno více než dvojnásobně.

Nepodařilo se detekovat všechny dopravní značky, ale nedetekovaných bylo relativně malé množství. Pokud byla využita i klasifikace dopravních značek, tak výsledky byly již horší, ale dle mého názoru by šlo dosáhnout podstatně lepších výsledků, pokud by byly přetrénovány modely pro klasifikaci dopravních značek.

Dopravní značky, které nebyly detekovány se dělí zejména na tři druhy. Jedna kategorie nedetekcí dopravních značek je způsobena příliš špatnými světelnými podmínkami ve kterých i upravený Cannyho detektor hran selže a nedetekuje všechny hrany dopravní značky. Druhá podstatná část dopravních značek, které nebyly detekovány jsou ty, které jsou z velké části překryty jiným objektem. Detekce dopravních značek se dokáže vypořádat s nepatrným překrytím, ovšem pokud je překrytí příliš velké, tak již dopravní značka není

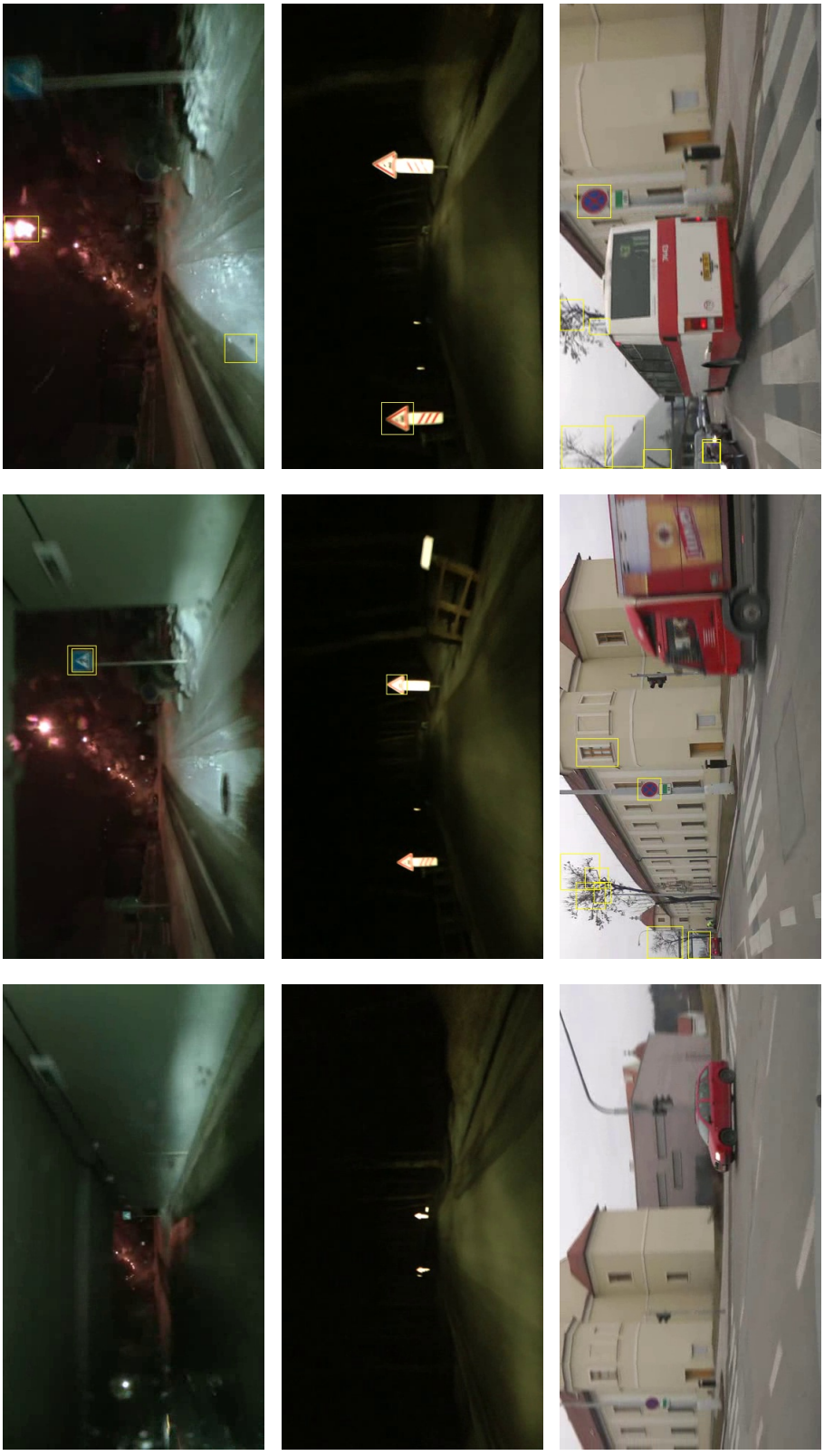
(a) Detekované dopravní značky

(b) Nedetekované dopravní značky

Obrázek 6.4: Ukázka detekovananých dopravních a nedetekovaných dopravních značek

detekována. Třetí kategorií dopravních značek, jež nebyly detekovány jsou ty, které jsou příliš rozmazané. Příklady detekovaných a nedetekovaných dopravních značek lze nalézt na obrázku 6.4.

Během vyhodnocování jsem také zkoušel, zdali je každá dopravní značka alespoň jednou detekována. Pro toto testování jsem vytvořil videa, která jsem sledoval zkoumal, bude-li každá značka detekována. Pro toto vyhodnocení byla použita videa z rozdílných světelných podmínek. Ukázalo se, že na těchto videích byla každá značka alespoň jednou detekována a ukázky těchto detekcí lze nalézt na obrázku 6.5.



Obrázek 6.5: Ukázky zpracování snímků z videa

Kapitola 7

Závěr

V mé práci jsem se zaměřil zejména na detekci dopravních značek v obraze, přičemž mým cílem bylo zpracovávat videa v reálném čase. Tento cíl se mi podařilo splnit, jak lze vidět v tabulce 6.3, která prezentuje, že jsem při detekci dopravních značek ve videu dosáhl rychlosti 66 snímků za vteřinu. Detekce je relativně úspěšná a pouze 6 % dopravních značek není detekováno.

Výsledný program, který byl v rámci mé bakalářské práce vytvořen, umožňuje také klasifikaci dopravních značek s využitím klasifikátoru převzatého z dimplomové práce [16].

Během vypracovávání jsem prostudoval různé způsoby detekce dopravních značek a tyto postupy jsem popsal v kapitole 2. Na základě výsledků, kterých bylo dosaženo pomocí daných algoritmů jsem vybral způsob, který jsem považoval za nejvhodnější a tento způsob detekce dopravních značek jsem implementoval.

Během průběhu implementace jsem prováděl experimenty s tímto systémem detekce dopravních značek a ten jsem dále vylepšoval, aby bylo dosaženo co nejlepších výsledků. Provedl jsem také konečné vyhodnocení systému, které bylo popsáno v kapitole 6.

Pro prezentování mé práce jsem vytvořil plakátek, jehož ukázka je uvedena v příloze C. O mé práci jsem také napsal článek, který jsem přihlásil do soutěže Student EEICT a tento článek byl přijat a zveřejněn ve sborníku této soutěže. V uvedené soutěži jsem se umístil v kategorii „Zpracování signálů, obrazu a biomedicína“ na 3. místě.

Další rozvoj mé práce by se mohl ubírat především dvěma směry. Lze využít například grafických karet pro obecné výpočty a implementovat systém s vysokou mírou paralelismu a tak dále urychlit celou detekci, případně i klasifikaci, dopravních značek. Nebo lze pro tento systém vytvořit modely pro klasifikaci dalších dopravních značek a provést testování v reálném provozu.

Literatura

[1] BARÓ, X.; ESCALERA, S.; VITRIÀ, J.; aj.: Traffic sign recognition using evolutionary adaboost detection and forest-ECOC classification. *IEEE Transactions on Intelligent Transportation Systems*, ročník 10, č. 1, Březen 2009: s. 113–126, ISSN 1524-9050.

[2] BAY, H.; TUYTELAARS, T.; GOOL, L. V.: SURF: Speeded Up Robust Features. V *Computer Vision – ECCV 2006*, *Lecture Notes in Computer Science*, ročník 3951, editace A. Leonardis; H. Bischof; A. Pinz, Springer Berlin / Heidelberg, 2006, ISBN 978-3-540-33832-1, s. 404–417.

[3] CANNY, J.: A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 8, č. 6, Červen 1986: s. 679–698, ISSN 0162-8828.

[4] DUDA, R. O.; HART, P. E.: Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, ročník 15, č. 1, 1972: s. 11–15, ISSN 0001-0782.

[5] FISHLER, M. A.; BOLLES, R. C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, ročník 24, č. 6, Červen 1981: s. 381–395, ISSN 0001-0782.

[6] FORSYTH, D. A.; PONCE, J.: *Computer Vision: A Modern Approach*. Prentice Hall, první vydání, Srpen 2002, ISBN 0-130-85198-1.

[7] FREEMAN, H.: On the Encoding of Arbitrary Geometric Configurations. *IEEE Transactions on Electronic Computers*, , č. 10, 1961: s. 260–268, ISSN 0367-9950.

[8] GARCIA-GARRIDO, M. A.; SOTELO, M. A.; Martin-Gorostiza, E.: Fast traffic sign detection and recognition under changing lighting conditions. V *Intelligent Transportation Systems Conference*, 2006, s. 811–816.

[9] GOEDEMÉ, T.: Traffic Sign Recognition with Constellations of Visual Words. V *5th International Conference on Informatics in Control*, 2008.

[10] MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H.: *Introduction to Information Retrieval*. Cambridge University Press, 2008, ISBN 05-218-6571-9.

[11] The Mobile Robot Programming Toolkit. [online]. [cit. 2012-03-18]. Dostupné z: `http://www.mrpt.org/`

[12] ROSENFELD, A.: Connectivity in Digital Pictures. *Journal of the ACM*, ročník 17, č. 1, Leden 1970: s. 146–160, ISSN 0004-5411.

[13] SHAPIRO, L. G.; STOCKMAN, G. C.: *Computer Vision*. Prentice Hall, Leden 2001, ISBN 0-130-30796-3.

[14] SONKA, M.; HLAVAC, V.; BOYLE, R.: *Image processing, analysis, and machine vision*. PWS Publishing, druhé vydání, 1998, ISBN 0-534-95393-X.

[15] SUZUKI, S.; ABE, K.: Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, ročník 30, č. 1, Duben 1985: s. 32–46, ISSN 0734-189X.

[16] SVOBODA, T.: *Detekce, lokalizace a rozpoznání dopravních značek*. Diplomová práce, FIT VUT v Brně, 2011.

[17] SZELISKI, R.: *Computer Vision: Algorithms and Application*. Springer, 2011, ISBN 978-1-84882-934-3.

[18] VIOLA, P.; JONES, M.: Rapid Object Detection Using a Boosted Cascade of Simple Features. V *Computer Vision and Pattern Recognition*, 2001, s. 511–518.

[19] WESOLKOWSKI, S.; JERNIGAN, E.: Color Edge Detection in RGB Using Jointly Euclidean Distance and Vector Angle. V *Vision Interface*, 1999, s. 9–16.

[20] ŽÁRA, J.; BENEŠ, B.; SOCHOR, J.; aj.: *Moderní počítačová grafika*. Computer Press, druhé vydání, 2004, ISBN 80-251-0454-0.

Seznam příloh

Příloha A

Obsah DVD

K mé práci jsou přiloženy dvě DVD, jež obsahují zdrojové kódy systému implementovaného v rámci mé práce a také vygenerovaná videa a obrázky. Tato DVD obsahují následující složky.

- **Poster** Tato složka obsahuje plakát vytvořený pro prezentování mé práce ve formátech PDF a SVG.
- **Report** Složka obsahující zdrojový text tohoto dokumentu a také jeho verzi ve formátu PDF.
- **Scripts** Zde se nachází podpůrné nástroje a skripty určené zejména pro vyhodnocování programu. Pro jejich použití bude velmi často nutné pozměnit cesty na začátcích těchto souborů.
- **SourceCodes** Tato složka obsahuje zdrojové kódy k implementovanému systému detekce a následné klasifikace dopravních značek. V této složce se také nachází program pro anotaci dopravních značek v obraze. Postup překladu těchto programů lze nalézt v příloze B a také v souboru README v této složce.
- **Videos** Výstupy videí jsou uloženy v této složce. Tyto výstupy jsou rozděleny do podsložek, přičemž v jedné jsou výstupy detekce a ve druhé i následné klasifikace. Klasifikace těchto dopravních značek byla prováděna pouze pro 5 natrénovaných modelů, takže ne všechny dopravní značky mohou býti v současné době klasifikovány. Videa umístěná na tomto DVD jsou často velmi rozdílná v podmínkách ve kterých zpracování probíhá. V některých situacích jsou dopravní značky velmi dobře vidět, jelikož videa jsou pořízena za dne. Na jiných dopravní značky nejsou natolik zřetelné, protože jsou pořízené za tmy a za špatné viditelnosti.
- **Photos** Detekované a klasifikované dopravní značky zakreslené do obrázků. Tato složka se nachází na DVD, které je označeno číslem 2. Složka obsahuje také ukázky XML souborů generované pomocí programu implementovaného v rámci mé bakalářské práce.

Příloha B

Překlad programu

Tato kapitola popisuje postup nutný pro překlad programu implementovaného v rámci mé bakalářské práce. Popis použití tohoto programu `SignsDetector` lze získat pomocí parametrů `-h` nebo `--help` nebo jej lze také nalézt v souboru README.

Pro překlad programu je nutné mít nainstalovanou knihovnu OpenCV[1], přičemž je vyžadována verze 2.3.1 a vyšší.

Pro přeložení programu je využíváno systému CMake[2], jenž umožňuje překlad na více platformách. Překlad programu lze realizovat pomocí následujících příkazů, přičemž tento příklad je uveden pro operační systém Linux. Složka uvedená na řádku 1 může být jiná v závislosti na aktuálním umístění zdrojových souborů programu. Pokud by překlad byl spouštěn se starší verzí knihovny OpenCV, tak vás upozorní spuštění příkazu `cmake`, jenž je uvedeno na řádku 2. Po úspěšném vykonání těchto příkazů budou spustitelné soubory umístěné ve složce `bin`.

```
1  cd SourceCodes
2  cmake .
3  make
```

[1] http://opencv.willowgarage.com/
[2] http://www.cmake.org/

Příloha C

Plakát